

ISECON '85

INFORMATION SYSTEMS
EDUCATION CONFERENCE

Proceedings

Fourth Annual

Information

Systems

Education

Conference

**The Conference Designed
For Academics In
Computer Information Systems**

SHERATON-HOUSTON HOTEL

Houston, Texas

OCTOBER 26-27, 1985

SPONSORED BY:

Data Processing Management Association
Education Foundation



ISECON
85

Proceedings
Fourth Annual
**Information
Systems
Education
Conference**

**The Conference Designed
For Academics In
Computer Information Systems**

**SHERATON-HOUSTON HOTEL
Houston, Texas
OCTOBER 26-27, 1985**



Sponsored by
**DATA PROCESSING MANAGEMENT ASSOCIATION
EDUCATION FOUNDATION**



Copyright and reprint permissions:

Abstracting is permitted with proper credit to the source. Instructors are permitted to photocopy isolated articles for noncommercial classroom use without fee. For other copying, reprint or republication permission contact:

Data Processing Management Association
505 Busse Highway
Park Ridge, IL 60068

WELCOMING REMARKS

Bruce E. Spiro, CDP
President, DPMA Education Foundation

Senior Consultant
Advanced Information Management, Inc.
1988 Opitz Blvd., Woodbridge, Virginia 22191

Welcome to ISECON '85. It is a fine comment on the concern and dedication of the computer educational professional that you have taken the time to be here. We are fortunate to be in a discipline that continues to grow and change. There is no room for the complacent or the static. For us change is, in fact, a way of life. It is to address this need for change and development that we are here. Computer information systems have become an integral part of business and industry. From an obscure and somewhat mystical art, just a few short years ago, information is now thought of as a resource, as much and as critical a resource as finances or personnel. This has been due in no small part to educators such as you who have perceived the ever increasing role that information plays and have devised the programs that define and illuminate.

I recall how we used to bemoan the fact that top management was not computer knowledgeable, and that so many people were not aware of how important information processing was. But, no more. This generation of top management is computer knowledgeable. They must be if the company is to survive and prosper. The computer is no longer a mystery. It forms an important part of the business plan. Along with this change in the environment it is critical that education must change and must be the leader of change. While fundamentals remain constant the depth of knowledge must increase. In the past, those lacking knowledge could make their point by foot stomping and arm waving, but now logic and reason are paramount. The audience is no longer either awed or bored. The challenge to the educator is tremendous. Not only to keep up with the dynamic developments, but to retain the leadership position.

It is important that education is at the leading edge of development in computer information systems for it is difficult enough for the computer professional to keep pace with such a dynamic industry without being taught the obsolete. The shelf life of a degree in high technology is shrinking without a doubt. Some say that there is a complete turnover of computer information systems knowledge every four or five years. All agree that updating of educational programs is a continuing and essential need. Unlike many disciplines we never get the chance to sit back and say "Look what I have done." For if we do, we quickly find that the world has passed us by. Within one generation of man the computer has blazed through four generations and is well along the fifth. That is real progress, and that is what makes computer information systems the interesting, challenging field that is. I certainly don't want it to be any different.

At the other end of the scale, we worried not too long ago about computer literacy. We were concerned that those entering school would not know what a computer is and that much time would be lost in bringing them up to speed. While this is still a concern in some areas we are now also seeing the reverse. Many children have a computer at home and in some areas are more skilled and knowledgeable than their teachers. But, we must be careful not to confuse the ability to use with understanding. Once more the challenge to the educator is great, for the understanding of computer information systems is the fulcrum that will leverage us into the future. It is you who will develop that understanding throughout the academic life of your students.

This level of knowledge at an early age brings with it considerable obligations to the educator; for knowledge is outrunning the wisdom to employ it. Moreover we don't have generations to develop the social awareness that will lead to moral constraints. Legislatures struggle to deal with the problems; so far without great success. The burden, once again, falls to the high technology educator. Along with knowledge we must impart social values, and moral standards. If we are to be leaders we must know the value and importance of what we do, recognize the impact of misuse, and appreciate the obligation of knowledge. And we must have the ability to pass these values on to the students.

The Data Processing Management Association formed the Education Foundation a few years ago in recognition of the critical part that you

play in the development of the most important and pervasive industry in the world today. From space exploration to toasters, there are few things that are not based on computer applications. From the banking industry to national defense, information and the processing of information is the life blood of effective performance. The need for knowledgeable professionals becomes greater each day. Along with this, the difficulty and challenge in education becomes greater.

That is why we are here: to accept this challenge; to take a hard look at how our computer information systems education programs are structured; how they can be improved; and to share our knowledge of what is good and what is not; what works and what doesn't.

I would like to thank the many people who have worked so hard to put this conference together. Particularly Dr. George Fowler, Conference/Program Chairman; Doris Palmer Luttrell, Marketing Chairman; Dr. John McKinney, Exhibitor Chairman; Dr. Jan Prickett, Facilities Chairman; Dr. Herbert Rebhun, Activities Chairman; and DPMA Headquarters staff member Hildegard Klemm, who has done so much more than just what she was asked. Another thank you is in order to IBM, whose very generous grant has made it possible to keep the cost of this conference within the range of all. The continuing support of a major company such as IBM is indeed gratifying.

There is an excellent program ahead of us. Let's get started with it.

ISECON '85

EXECUTIVE COMMITTEE

**CONFERENCE CHAIRMAN
AND
PROGRAM CHAIRMAN**

GEORGE C. FOWLER

Associate Professor of MIS
Texas A&M University

MARKETING CHAIRMAN

DORIS LUTRELL

Market Data Analyst
Drackett Co.

EXHIBITOR CHAIRMAN

JOHN MCKINNEY

Associate Professor of QA/IS
University of Cincinnati

FACILITIES CHAIRMAN

JAN PRICKETT

Assistant Professor of MIS
Northern Kentucky University

ACTIVITIES CHAIRMAN

HERBERT REBHUN

Associate Professor of MIS
University of Houston

TABLE OF CONTENTS

WELCOMING REMARKS	i
ISECON '85 EXECUTIVE COMMITTEE	ii
KEYNOTE ADDRESS	
Information System Development: Theory and Practice	1
Harlan Mills	
CIS CURRICULUM	
DPMA Graduate Model Curriculum in Management Information Systems. A Preliminary Report	8
Doris G. Duncan	
Associate Degree Model Curriculum for: Computer Information Systems	11
Rod B. Southworth	
COBOL 8X - Teaching the New Compiler	13
Robert T. Grauer	
An Evaluation of Information Systems Methodologies: Self Study vs. Lecture	16
George W. Morgan, Wayne Headrick and Walter E. Johnston	
A Team Project Approach to Teaching Systems Analysis	18
Jean Buddington Martin	
The Live Student Systems Development Project	20
Iza Goroff	
The Role of Microcomputers in the CIS Curriculum.....	24
Wayne D. Smith	
Developing Micro-Based Courses An Information Systems Model	26
Thomas E. Burke	
Profile of Introductory Database Courses Offered at AACSB-Accredited Institutions	29
Joan K. Pierson, Jack Shorter, Jeretta A. Horn and G. Daryl Nord	

The DPMA Model Curriculum and AACSB Accreditation	33
Alden C. Lorents	
CIS Curriculum in a Two-Year Program	36
Marvin Gore	
CIS Curriculum - The Canadian Academic Perspective	47
Maria A. Fine	
* DPMA Curriculum Development Process	
Don Price	
The UW-Whitewater Management Computer Systems Program: A Curriculum Adapted to Change	49
Jacob Gerlach, Iza Goroff and Robert Horton	
Graduate Education for Training Information Systems Concept	51
John F. Schrage and James Savage	
An Undergraduate CIS Program Within a Computer Science Department	54
Ali Behforooz and Onkar P. Sharma	
The Agony and Ecstasy of Teaching DSS	58
Kuriakose Athappilly and William Leigh	
Teaching CIS-15, Information Resource Management	62
Sadra E. Poindexter	
Teaching Structured COBOL Programming Using a Model	65
Ron Teemley	
Curriculum for Introduction to Computer-Based Systems A Prospective for the Future	69
Enming Lin, Donald E. Carr and Chang-Yang Lin	
An Industrial Survey of End User Computing Topics	72
Robert L. Horton	
The Information Processing System Model: A Conceptual Model for CIS Instruction	75
James A. O'Brien	
* Office Systems Model Curriculum	
Kathy White	

FUTURE TRENDS IN INFORMATION SYSTEMS

The Development of an Expert System to Assess the Effects of Computer Maintenance Service	79
Avi Rushinek	

Intelligent Strategy Planning System	82
Y. Dolly Hwang	
BASS: Business Application and Support Systems	84
L. A. Adams and M. A. Bassiouni	
Future Computer Languages	88
Jagtar S. Chaudhry and Nicklaus Damachi	
Developing Effective Management Information End Users	90
Patricia C. Eiland	
The Impact of Electronic Mail Systems on Managerial and Organizational Communications	93
Mary Sumner	
Electronic Word Processing and Office Automation: A Systems View	96
Madjid Tavana	
Knowledge Engineering Software Supporting Delivery of an Expert System Within the CIS Curriculum	98
James H. Blaisdell	
Using a Microcomputer DBMS to Teach Relational Database Theory and Practice	102
David L. Russell	
The Impact of User Developed Applications on Systems Development	107
Mary Sumner	
The Psychology of Effective Prompting	110
Gerald N. Pitts	
Overview of the Fourth Generation Languages Movement with Academic Use Integrated	112
Boulton B. Miller and John F. Schrage	
Effective End Users and Fourth Generation Tools	116
Carol Chrisman and Barbara Beccue	
Tutorial: Expert Systems	118
James H. Blaisdell	

BUSINESS AND SOCIAL ISSUES

The Matrix Life Cycle: An Integrated Model of the Organizational Dynamics of Implementation	120
Dennis P. Heaton	
Establishing an Executive Advisory Committee: A Case Study	123
Khris McAlister and M. Blaine McCormick	

Strategic Information System as the Output of an Open System	125
Madjid Tavana	
Introduction of a Computerized Management Information System in a Non-Sophisticated Environment	128
Tom Marshall, Bruce L. McManis, Connie Schulte and L. W. Shell	
An Industrial Computer Literacy Program	130
Connie D. Lightfoot	
High Technology, Employee Displacement and Social Responsibility	132
Michael Bisesi	

DELIVERY SYSTEMS AND PRODUCTIVITY

The Design and Analysis of the Information Systems Curriculum	135
Karen A. Forch	
Information Systems Needs for the Business Curriculum	138
John F. Schrage and Robert A. Schultheis	
Selective Guidelines for Effective Courseware Authoring	141
Rick David Stuart	
The Kodalky Folksong System	143
Iza Goroff and Robert Perinchief	
A Technique for Individualizing Machine Scored Tests	146
Richard K. Brewer	
Conducting Computer Literacy Workshops for Faculty Colleagues: The Experience of Two Workshop Leaders	148
Kathleen Duffy and Paula Mitchell	

STUDENT PAPERS

The Future Role of Microcomputers in Auditing: An Expert System Explored	150
Debbie Dare	
The Psychological Effects of Automation on Office Workers	154
Karen A. Lensler	
Microcomputers: The Growing Need and Application in Special Education	157
Patricia Miller	

A NEW COURSE IN INFORMATION SYSTEMS DEVELOPMENT

Harlan D. Mills
IBM Corporation
Bethesda, Maryland

A new course in information systems development is being taught in the College of Business and Management, University of Maryland. The course is based on the mathematical principles of system structure that underlie the curriculum of the IBM Software Engineering Institute, further applied to business processes and information systems. These principles extend the methodology of software development by structured programming to a methodology of system development by box structured analysis and design.

The course teaches principles that underlie effective analysis and design procedures needed in information systems development. It does not teach appearances. In illustration, we can teach trick dogs the appearances of doing arithmetic, but they cannot apply these appearances in any other situations than taught. But, when we teach children the principles of arithmetic, they can apply these principles to solve new problems never seen before. As a result, the course emphasizes exercises and examinations, rather than projects. As with children in arithmetic, we expect students to have no difficulty in applying these principles in real projects and situations. In fact, several students have credited this course with getting them better jobs than they expected because they could discuss the principles of their work in interviews.

The content of the course has been developed by A. R. Hevner, R. C. Linger, and H. D. Mills, and has been taught by Professor Hevner and another instructor to over two-hundred students, at both graduate and undergraduate levels. Student response has been surprisingly positive, even though the material is quite exacting in its logic and unconventional for information systems development courses. Students soon learn and appreciate the value of principles, rather than appearances, in their own careers, even in the short term. If they know the principles, they will have little difficulty in looking like systems analysts and designers, and they know that, too.

The students also seem to appreciate the emphasis on logical precision in the course, even though we all know that the deepest and most persistent problems of information systems finally came down to dealing with people. These people problems are difficult at best, but are made even more difficult or impossible by poorly addressed logic problems. For example, if a bank has a faulty accounting process that makes many errors, irate customers and overworked employees may look like people problems. But the way to address them is to remove the cause of the faulty logic, not by increasing customer relations personnel. The university is the place for students to get the logic problems of information systems development out of the way. They will be working on the people problems their whole careers.

These principles are described briefly in the following excerpts from a technical report by A. R. Hevner, R.C. Linger, and H. D. Mills.

1. A Historical Perspective

The introduction of computer technology into business operations brings the potential for more management control in administrative and analytical phases of business. But the rapid, almost pell-mell, introduction of computer technology of the past thirty years has sometimes brought a net loss of real management control because of a necessary dependence on personnel more versed in computers than in business operations. On top of that, the explosive growth of the computer industry itself has created problems of its own in a haphazardly educated and poorly disciplined work force that has difficulty in meeting schedule, cost, and reliability targets in information systems development.

Thirty years ago there was no such thing as the database systems, management information systems, and decision support systems that dot the information systems landscape today. Even so, it is sobering to reflect how short thirty years is in terms of intellectual development. When civil engineering was thirty years old, the right triangle was yet to be invented; when accounting was thirty years old, double entry principles were unknown.

To be sure, many more people are working on information systems than were working in civil engineering or accounting in their first thirty years. But fundamental ideas and deep simplicities take time. Even with all the excitement and progress there is still a lot to discover--possibly the right triangle for information systems.

The structured revolution [Yourdon] that changed cut and try computer programming to software engineering was triggered by a new concept called structured programming. Structured programming cleared a control flow jungle that had grown unchecked in dealing with more and more complex software problems for twenty years. It replaced that control flow jungle with the astonishing assertion that software of any complexity whatsoever could be designed with just three basic control structures, sequence, alternation, and iteration, which could be nested over and over in a hierarchical structure (the structure of structured programming). The benefits of structured programming to the management of large projects are immediate. The work can be structured and progress measured in top down development in a direct way. Properly done, when a top down development is 90% done, there is only 10% left to do (in contrast to projects which at 90% done often required another 90% to complete).

Structured programming has a rigorous mathematical foundation which can be used for management advantage. First of all, a so-called Structure Theorem establishes that any flow chart program can be designed as a structured program [Linger]. Therefore, a management standard of structured programming is technically sound. Second, a Top Down Corollary to the Structure Theorem establishes that a structured program can be created in a top down sequence such that each line can be verified correct by reference to previous lines (and not to lines yet to be created). This means that software can be created correctly as it is developed, without a final and unpredictable stage of trying to make it all work together.

The management benefits begin with standard practices for software development that are based on this mathematical foundation. Software personnel can be uniformly educated to these practices, with improved management visibility into the development process and improved communication between programmers in both the design and inspection phases. As a result, large-scale software projects previously risky or impossible can be completed consistently within schedules and budgets. For example, top down structured programming has been used extensively in the U.S. space shuttle program [Madden]; it is safe to say that the shuttle could not be flying (orbiting) today without structured programming.

Even so, business information systems development is more than software development. The operations of business involve all kinds of data, stored and processed in all kinds of ways. A simple encyclopedic description of such data and its uses leads to a data flow jungle that is even more tangled and arcane than the control flow jungle. Once again mathematics and engineering have come to the rescue by replacing the data flow jungle with just three basic system structures that can be nested over and over in hierarchical system structure. These system structures are called black box, state machine, and clear box; they provide a disciplined way to specify, design, and implement information systems and their subsystems to every level of detail. The data flow becomes a by-product of the methodology and now takes its structure from the system, not as an end in itself.

Next, in Section 2 we introduce the basic concepts and principles of box structures in system analysis and design. Section 3 illustrates the use of these basic concepts in an example of analysis of a U. S. Navy Supply System reorder policy. In Section 4 these ideas are used to construct a management methodology for information system development. In Section 5 we illustrate the Box Structure Methodology in an example of design in The New York Times Information Bank.

2. Box Structures of Information Systems

2.1 Box Structures

The three basic system structures are the black box, state machine, and clear box. They are three views of the same information system or any of its subsystems.

The black box gives an external view of a system or subsystem that accepts stimuli, and for each stimulus produces a response before accepting the next stimulus. A diagram of a black box is shown in Figure 1. The "system" could be a calculator, a computer, or even a manual work procedure that accepts stimuli from the environment and produces responses one by one. As the name implies, a black box description of a system omits all details of internal structure and operations, and deals solely with the behavior that is visible to its user in terms of stimuli and responses. Any black box response is uniquely determined by the system's initial conditions and stimuli history.

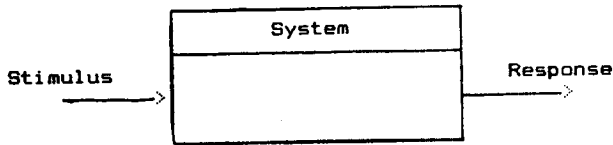


Figure 1. A Black Box Diagram

The state machine gives an intermediate system view that defines an internal system state, namely the data stored from stimulus to stimulus. Every black box has a state machine description. A state machine diagram is shown in Figure 2.

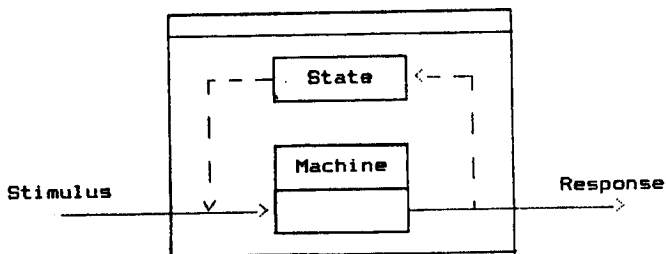


Figure 2. A State Machine Diagram

The state machine part called Machine is a black box that accepts as its stimulus both the external stimulus and the internal state and produces as a response both the external response and a new internal state which replaces the old state. The role of the state machine is to open up the black box description of a system one step by making its data visible.

The clear box, as the name suggests, opens up the state machine description of a system one more step in an internal view that describes the system processing of a stimulus and state (stored date). The processing is described in terms of the three control constructs of structured programming, namely sequence, alternation, and iteration, as shown in Figure 3.

Machine parts M1, M2 are black boxes; each accepts as its stimulus both a stimulus and state and produces as its response both a response and a new state. For example, in the Sequence Structure the clear box stimulus is the stimulus to black box M1, whose response becomes the stimulus to M2, whose response is the response of the clear box. Machine part C is a conditional switch that accepts a stimulus and a state, and then transmits that stimulus to one of two other parts but does not affect the state. For example, in the alternation structure, conditional switch C transmits the stimulus to either M1 or M2, while in the iteration structure, C transmits the stimulus to either M1 or the next part of the next higher structure. Nested hierarchies of these control constructs provide the capability to design all types of system processing.

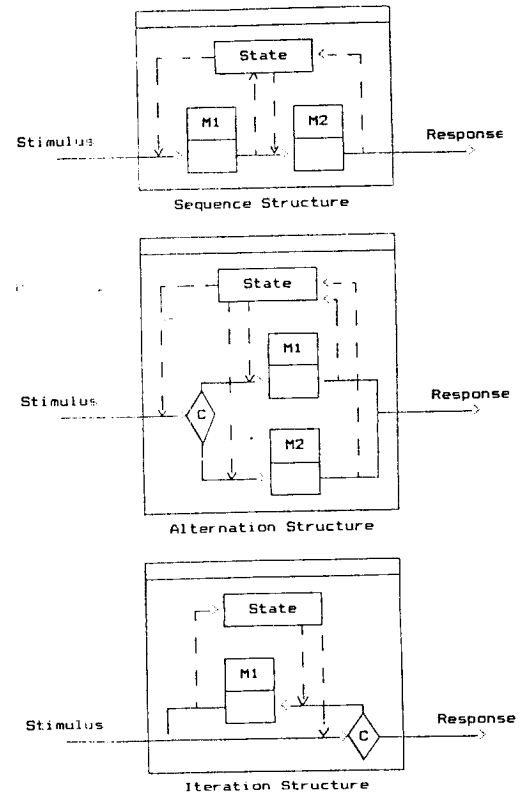


Figure 3. Clear Box Diagrams
(Mi = Machines)

At this point, a hierarchical, top-down description can be repeated for each of the embedded black boxes at the next lower level of description. Each black box is described by a state machine, then by a clear box containing even smaller black boxes, and so on.

These views represent an increasing order of internal system detail. The black box describes the system from a user view. The user view is foremost since the objective of business systems is to provide user services. The state machine adds the consideration of system data (State) and its manipulation (Machine). The clear box completes the description by adding internal processing details and recognizing embedded subsystems. Describing each subsystem in these increasingly detailed views provides an internal consistency that is essential in developing and managing systems. A system developer is forced to be consistent: the data structure must be consistent with the user view and the processing structure must be consistent with the data structure. System management is helped by the thorough documentation of the mappings between the system views.

2.2 Box Structures in Business Operations

Although the concept of a box structured hierarchical system is easy to see, its use in actual business systems requires business knowledge as well as computer knowledge. In fact, the box structures provide a form in which to describe business knowledge in a standard way. The principal value of a black box is that any business information system or subsystem will behave as a black box whether consciously described as such or not. In turn, any black box can be described as a state machine (actually in many ways), and any state machine can be described as a clear box (also in many ways), possibly using other black boxes. In practice, information systems or subsystems often have their own natural descriptions that can be reformulated as box structures.

In illustration, a 12 months running average defines a simple, low-level black box that might be used in sales forecasting. A stimulus of last month's sales of an item would produce a response of the past year's average monthly sales of the item; each month a new sales amount produces a new average of the last 12 months. Figure 4 shows the running average black box diagram. From the stimulus history, the black box transition formula for the response R at the end of month m is

$$R(m) := \frac{S(m) + S(m-1) + \dots + S(m-11)}{12}$$

Where the $:=$ symbol means that $R(m)$ on the left side is given the value of the expression in known values $S(m), \dots, S(m-1)$ on the right side.

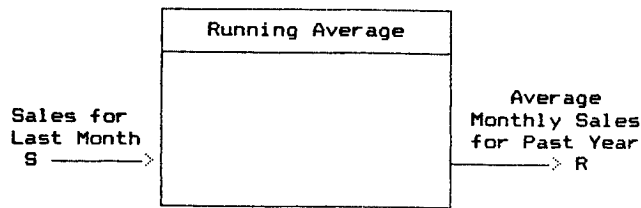


Figure 4. Running Average Black Box

A possible state machine of this black box would identify that the previous 12 monthly sales are to be stored in the state. The machine, upon receiving the last month's sales, would update the state by discarding the oldest sales value and storing the input sales value, then calculate the new running average response. Figure 5 shows the state machine diagram.

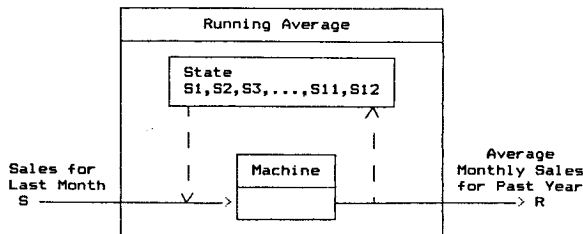


Figure 5. Running Average State Machine

A clear box will describe how the state updating process and the averaging process are performed. One possible design is shown in Figure 6. The Update State and Find Average machines are simple enough to include their processing details directly in a sequence structure. In this case, no further black box description is needed because neither Update State nor Find Average requires any state data themselves.

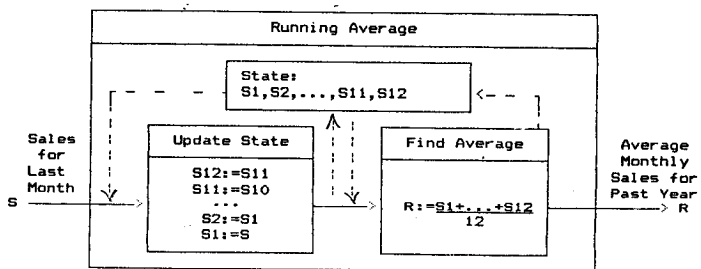


Figure 6. Running Average Clear Box

Note that many other state machine and clear box designs could have been chosen to implement the running average black box. For example, the state data could be stored as monthly sales values divided by 12. Then, the running average would be found by adding all the state data.

A running average black box is a simple sales forecaster. However, if sales are seasonal or have definite trends, a more suitable black box is required. Such a forecaster will differ in details, but can still be described in a black box/state machine/clear box structure.

If a human forecaster is known to be successful, it may be of interest to incorporate that wisdom into a forecasting black box for an entire inventory, e.g., for 10,000 items, which are beyond the ability of the human forecaster to deal with one by one. In this case, the human forecaster may not be able to describe a black box behavior directly. Instead, the description may come out as a mental process of recollections and calculations that involve both state machine and clear box behavior. The box structure discipline gives a systematic basis for interviewing such a human forecaster and converting that human wisdom into systematic form. The result will be a forecasting black box/state machine/clear box structure that can be analyzed as part of a larger system, e.g., an inventory control system, with its own box structure.

3. The U. S. Navy Supply System Reorder Policy

3.1 Background

The creation of clear box descriptions out of existing business processes, and their conversions into black box descriptions can be useful directly. For example, in the middle fifties, an analysis in the U. S. Navy multiechelon supply system led to a radical revision and improvement in inventory control. The basis for this analysis was the conversion of a clear box description of inventory reordering into a state machine, then into a black box description. At the time, the current Navy Supply System reorder policy, called the "k months of supply" policy, seemed sensible enough. It called for maintaining some factor k times an average month's demand of an item either in inventory or on order. The factor k was chosen by the inventory manager to reflect the length of the pipeline, the variability of demand, and the consequences of outage for the item. This k varied for item to item, say from anchors to socks, but once chosen, the rest of the calculation of each month's reorder was simple and automatic. The average demand was calculated by a 12 months running average, so the effects of an unusual month would seem to be averaged out. For example, if the manufacturing cycle for making a certain size anchor is 9 months, a variation of 3 months demand could be expected and the consequences of outage indicate another 3 months safety factor, then k would be $1.5(9 + 3 + 3)$ months.

3.2 The Clear Box Formulation

The clear box can be formulated directly from the business process. The clear box description of the k months of supply policy for a particular item has as state data the value of k for the item, the current inventory (including items on order), and the past 12 months of demands. With the stimulus of last month's demand, the new state is obtained by discarding the oldest demand, retaining the current one and subtracting it from last month's inventory to get current inventory. Next, the running average of the past 12 months is computed, multiplied by k , and then the current inventory is subtracted to get the reorder value. Finally, inventory (which involves items on order) is increased by adding the reorder just calculated. This clear box is depicted next in Figure 7, using variables k, I (for inventory), $D_1, D_2, \dots, D_{11}, D_{12}$ for the past 12 months of demands; D is the current demand and R the resulting reorder.

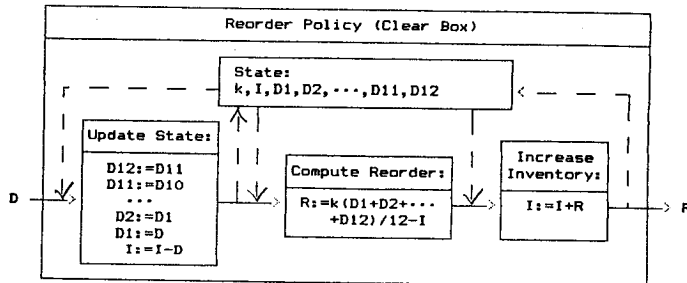


Figure 7. Reorder Policy Clear Box

This clear box description represents an actual business process developed on a pragmatic basis that seems to make a lot of sense. Once formulated, however, it can be converted rigorously into a state machine, then into a black box for further understanding and insight.

3.3 The State Machine Derivation

Next, the state machine can be determined by eliminating the clear box sequence structure by finding single expressions for the response R and each state variable $I, D_1, D_2, \dots, D_{12}$ in terms of the stimulus D and the last values of the state variables. On examination of the clear box, it can be seen that the new values of D_1, D_2, \dots, D_{12} are given by the Update State part because that is the only place they are updated. The expression for I can be determined from the two parts in which I is updated. In this case, D is subtracted from the last state value of I in Update State, then R is added to this intermediate value of $I - D$, so the new state value for I is

$$I := I - D + R.$$

However, R must be worked out before I is known in terms of the stimulus and old state. In this case, R is updated only in Compute Reorder in an expression that contains $D_1, D_2, \dots, D_{12}, I$ and D . But all these variables except D were just updated in Update State, which replaces D_1 by D, D_2 by D_1, \dots, D_{12} by D_{11} and I by $I - D$. Therefore, the expression

$$R := k(D1 + D2 + \dots + D12)/12 - I$$

in intermediate state data becomes

$$R := k(D + D1 + \dots + D11)/12 - (I - D)$$

in terms of the old state data. Now, I can be finally worked out, from

$$I := I - D + R$$

to

$$I := I - D + k(D + D1 + \dots + D11)/12 - (I - D)$$

and the last term $(I - D)$ cancels the first two terms, so I is simply

$$I := k(D + D1 + \dots + D11)/12.$$

At first glance, it may seem surprising that I is just k times the average of the last 12 months of demands, but on second thought, that is just what the k months of supply reorder policy should produce. The state machine so derived above is depicted in Figure 8.

In this case there are no real surprises in the Reorder Policy State Machine. But it has been distilled down one step by removing the sequential dependencies of the clear box. Note, however, that the so called "material balance" equation---that new inventory should equal old inventory plus additions minus deletions is automatically accounted for in this State Machine---it is not a required inspiration of an analyst to remember or account for it.

3.4 The Black Box Derivation

With sequential dependencies of the clear box eliminated to get the state machine, the next step is to eliminate the state dependencies of the state machine to get the black box. In doing so, it will be necessary to introduce previous demands into the single expression for the response. Let $D(m)$ be the demand for month m, the state data for the state machine that accepts stimulus $D(m)$ be $I(m), D1(m), D2(m), \dots, D12(m)$ and the response to this stimulus be $R(m)$. Now, the new state that will be updated from stimulus $D(m)$ will be $I(m+1), D1(m+1), D2(m+1), \dots, D12(m+1)$ for next month. Then, an inspection of the Reorder Policy State Machine of Figure 8 shows that the response and new state values will be as follows:

$$\begin{aligned} R(m) &= k(D(m) + D1(m) + \dots + D11(m))/12 - I(m) + D(m) \\ I(m+1) &= k(D(m) + D1(m) + \dots + D11(m))/12 \\ D12(m+1) &= D11(m) \\ D11(m+1) &= D10(m) \\ &\dots \\ D2(m+1) &= D1(m) \\ D1(m+1) &= D(m) \end{aligned}$$

Note that these are equations (=), rather than assignments (:=); the month indexes make this possible and correct. In particular, these equations hold for m replaced throughout any equations by another expression for m. For example, these values could be computed on a spreadsheet, with headings for the stimulus, response and state data and initial values for the state; then a column of stimuli values could produce the rest of the values of the spreadsheet automatically. More concretely, given initial values for

$$k, I(1), D1(1), D2(1), \dots, D11(1), D12(1)$$

for the first row of the spreadsheet and an input column of values under D, referred to as $D(1), D(2), \dots, D(m)$; the spreadsheet process will compute first $R(1)$, to complete the first row, then $I(2), D1(2), D2(2), \dots, D11(2)$, and $D12(2)$. The second demand, $D(2)$, would produce the second response, $R(2)$, and the state for the third iteration; and so on. Of course, all of this processing is done so rapidly that it is not noticed by the spreadsheet user. But, while some intuition could be obtained by trying various columns of stimuli, we will see, in this particular case, that a symbolic mathematical analysis of these equations will lead to a major revelation.

In order to carry out the elimination of state data from this set of equations, it turns out to be convenient first to express $D1, \dots, D12$ in terms of demands D. Since

$$D1(m+1) = D(m)$$

is an equation, replace m by m-1 on both sides to get

$$D1(m) = D(m-1)$$

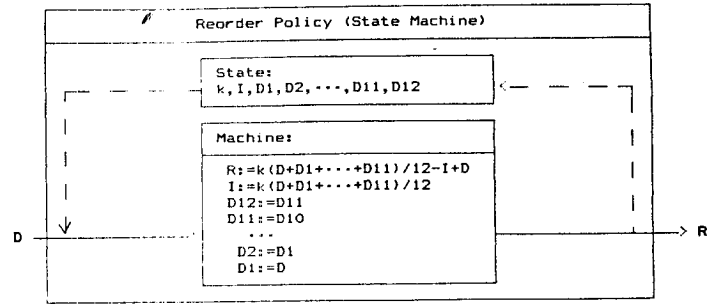


Figure 8. Reorder Policy State Machine

Next,

$$D2(m) = D1(m-1) = D(m-2)$$

...

$$D11(m) = D(m-11)$$

$$D12(m) = D(m-12)$$

as can be expected with a little thought. Now, both $R(m)$ and $I(m)$ can be expressed in terms of demands D instead of state data $D1, \dots, D12$, but it will be convenient, also, to substitute the expression for $I(m)$ (obtained by replacing m by m-1 throughout the equation for $I(m+1)$) in $R(m)$, to obtain

$$R(m) = k(D(m) + D(m-1) + \dots + D(m-11))/12 - k(D(m-1) + D(m-2) + \dots + D(m-12))/12 + D(m)$$

Now, the surprise is that 11 terms of the first line of the right side are exactly the same as 11 terms of the second line, but with opposite signs---they cancel out! Therefore, $R(m)$ reduces to

$$R(m) = (kI(m) - kD(m-12))/12 + D(m)$$

which simplifies to

$$R(m) = (1 + k/12)D(m) - (k/12)D(m-12).$$

That is, the Reorder Policy Black Box is given by a weighted combination of exactly two demands. The surprise is that $R(m)$ depends on only two demands $D(m)$ and $D(m-12)$, a year apart, even though a running average was used in defining $R(m)$ in its business process and clear box description. It just happens that the interactions of the material balance and the reorder policy cancels out the effect of all the intermediate demands. These interactions and cancellations would also be taking place, over and over, in spreadsheet calculations, but the chances of discovering such a pattern would be very remote. As evidence, this reorder policy had been in use by many organizations over many decades without any hint that such a pattern existed. That is, a lot of human thought and observation of results did not lead even to a suspicion of this pattern!

3.5 Using the Black Box Description for Further Analysis

Even though the form of the Reorder Policy Black Box is a surprise, is that bad? The Reorder Policy is used in a multiechelon hierarchy from many small supply points at the bottom up through a few large ones (ultimately a few suppliers, possibly only one) at the top. The objective of the Reorder Policy, beyond providing supplies, is to smooth, or dampen, out the demand variability necessarily expected at its bottom to get a more level aggregate of demands higher up in each echelon, so that the ordering to outside suppliers at the top is as level as possible. The effort of such smoothing through several echelons is multiplicative, and can be very effective. For example, if each echelon reduced the demand variability by a factor of two for example, then the effect through two echelons would reduce the variability by a factor of four and through three echelons, a factor of 8 over the variability at the bottom. In turn, steady orders on the outside suppliers can mean lower costs per unit because of the economics of stable production. That is, if the k months of supply policy, used throughout the multiechelon system smoothed demand variability at each reorder point, it could effect the economics of supply significantly. Now that the black box of the reorder policy has been derived, it is possible to analyze the smoothing of reorders from demands. The reorder R has the form (simplifying notation).

$$R = (1 + k/12)D - (k/12)D^*$$

where D is last month's demand and D^* is the demand a year ago. First of all, if demand is constant, say $D0$, then

$$\begin{aligned}
 R &= (1 + k/12)DO - (k/12)DO \\
 &= (1 + k/12 - k/12)DO \\
 &= DO
 \end{aligned}$$

so reorders will exactly match demands, a good thing because inventory will be completely stable. Now, consider the variability of demands D and D*. If D or D* are unusually high or low the other may compensate, or may not.

In order to develop a concrete numerical illustration, suppose k=12, so R has the especially simple form

$$R = (1 + 12/12)D - (12/12)D^* = 2D - D^*$$

Suppose that D and D* average 100 units, but are 75, 100, 125 each with an independent probability 1/3. Then there are 9 equally likely cases for (D, D*) values. For example, if D=75, D*=125, then

$$R = 2(75) - 125 = 25.$$

When these cases are listed the values of R are given in Table 1.

		D*:		
		75	100	125
D:	75	75	50	25
	100	125	100	75
	125	175	150	125

Table 1. Values for R

Surprisingly, Table 1 shows demands D and D* vary only at most 25 from their average value 100; the reorder R varies up to 75 from its average value of 100. In fact, Table 1 shows that the reorder policy does not dampen the variability of demands at all; it amplifies them---in this case up to a factor of 3! A more extensive statistical analysis verifies this illustration. The standard derivation of R turns out to be $\sqrt{5}$ (≈ 2.236 . . .) that of the standard deviation of D and D*. That is, the k months of supply policy is an inadvertent demand variability amplifier in the multiechelon supply system. Just as dampening is multiplicative so is amplification. Through 3 levels, rather than reducing variability by a factor of 8, this reorder policy in fact increases variability by a factor of $(\sqrt{5})^3 = 11.18$. . .

This clear box to black box analysis showed that most of the variability of inventory levels and reorders in the upper echelons of the Navy supply system was self induced by a seemingly sensible reordering policy. Once the problem was revealed, it was possible to devise a new kind of reordering policy, called an exponential smoothing policy, that reduced the variability of demands up the echelons rather than amplifying them.

At the time of this analysis of the Navy supply system, this clear box to black box analysis was novel and deep enough to be published in the professional mathematics and economics literature [Mills]. In the box structure methodology today, any information systems analyst should be capable of the analysis, but even more importantly, capable of initiating the analysis on a routine basis. That is, the box structure methodology defines standard analyses that previously have been considered ingenious and inspired. These standard analyses involve the formulation of business processes from current operations in clear box terms, then the derivation of the state machine and black box equivalents on a routine basis using standard principles and procedures.

4. Managing Information Systems Development

4.1 Box Structure Hierarchies

The small example of a running average illustrates the concepts but not the scope of box structures. Any business information system behaves as a black box for its users. They enter data (stimuli) and receive data (responses). Data entry may be by keystroke, by punched cards, even by automatic sensors such as optical scanners. Data output may be on computer displays, hard copy, even machine readable media. For example, an airline reservation clerk uses the reservation system as a black box. But inside is a gigantic state machine (the state is the data of the entire system) and a corresponding clear box (the system state and top level programs of the system).

A database system such as IMS behaves as a black box, with application programs as its users. The state machine can be visualized in storage and retrieval terms, while the clear box will be involved with storage and retrieval computation. In this case the information system using the data base system as a black box component will itself behave as a black box for its human users.

That is, business information systems and their subsystems all exhibit black box behavior, and thereby admit black box/state machine/clear box structures. As a result, identical structures and methods of reasoning can be used during information systems analysis and design in a hierarchy of smaller and smaller subsystems, as show in Figure 9.

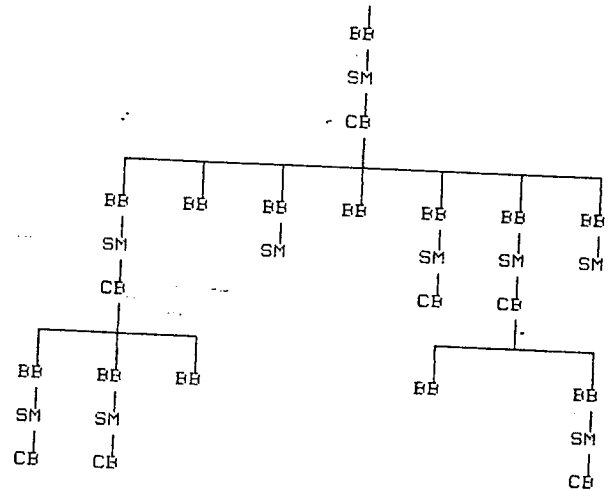


Figure 9. A Box Structure Hierarchy

A box structure hierarchy itself provides an effective means of management control over large, complex systems. By identifying black box subsystems in higher levels of the system, only a manageable amount of state and processing needs to be handled within each box structure.

Each subsystem becomes a well-defined, independent module in the overall system. Two important concepts in box structure hierarchy are black box replacement at any point of the hierarchy and state migration between points of the hierarchy. The concept of black box replacement is key in system development for the management flexibility it provides. A black box is a unit of description that can be isolated and is independent of its surroundings in a system. In particular, a black box can be redesigned as many different state machines and clear boxes. As long as the new black box behavior is identical to the original; the rest of the system will operate exactly as before. Such black box replacement may be required or desirable for purposes of better performance, changing hardware, or even changing from manual to automatic operations.

State migration through the box structure hierarchy is a powerful tool in managing system development. It permits the placement of state data at the most effective level for its use. Downward migration is possible when black boxes are identified in a higher level clear box; state data used solely within one black box can be migrated to the next lower level of the box structure hierarchy. The isolation of state data at its essential level in the system hierarchy provides important criteria for the design of database systems and file systems. Upward migration is possible when duplicate state data is updated in identical ways in several places in the hierarchy. This data can be migrated up to the closest common parent subsystem for consistent update at one location.

The box structure concepts provide a solid basis for the management and control of all development activities. New information, better ideas, even setbacks can be expected throughout information system development. The box structure hierarchy provides a framework for orderly control and process, rather than the chaos that such new information, better ideas, and setbacks can generate through many iterations. Black box replacement and state migration provide the necessary creative flexibility during the system development by allowing improvements in the design without losing its integrity.

4.2 Managing Information Systems Analysis and Design

The box structure of information systems leads to precise definition of the work of analysis and design, as shown below in Figure 10.

Given a black box/state machine/clear box structure for a system or subsystem, it is an analysis activity to deduce the black box from the state machine or to deduce the state machine from the clear box, while it is a design activity to induce a state machine from a black box or to induce a clear box from a state machine. The design activity does not produce a unique product because there are many state machines that behave like a given black box, and many clear boxes that behave like a given state machine. The analysis activity does produce a unique product because there is only one black box that behaves like a given state machine and only one state machine that behaves like a given clear box.

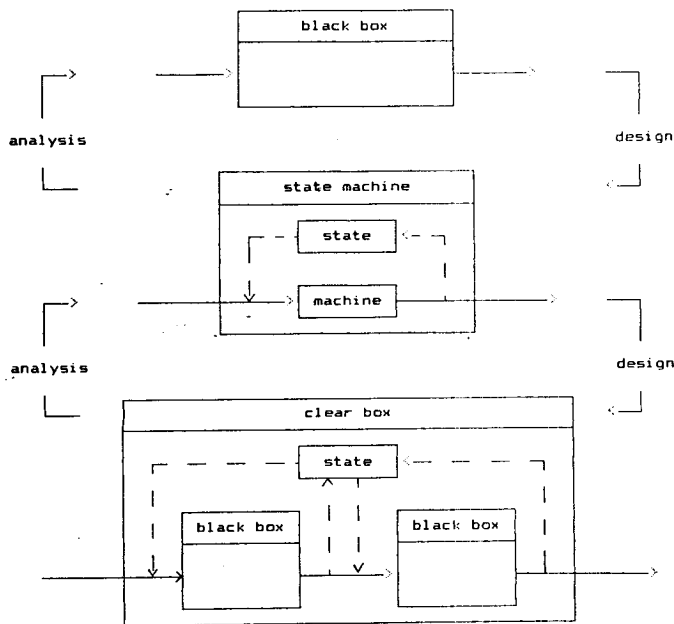


Figure 10. Analysis and Design

These definitions of analysis and design allow work assignments and reporting to be precise and comprehensive in managing information systems development. Each box structure analysis or design step represents a discrete unit of work, which altogether create the analysis and design of an entire system. The analysis activities of manual procedure reviews and interviews fit directly into these precise definitions. A person interviewed will describe procedures that the analyst will formulate as a clear box, then convert into state machine and black box terms. The design will then proceed from the derived black box back through a state machine and clear box better suited for automatic processing.

Information system development may take months or years, and require from a few to a few dozen, even a few hundred, people. Each of these people are discovering new facts about the business or the system, identifying new problems and old problems, finding solutions to those problems, making logical decisions about data storage and processing, and so on, every day. Even a small information system involves a large amount of logical structure and detail in its development. It is imperative to keep all this structure and detail organized and accessible for the developers in the conduct of the work.

The hierarchical box structure of black boxes, state machines, and clear boxes is designed explicitly to keep the details of analysis and design accessible during information system development. But there still must be a physical medium for recording this structure and its details. For this reason, this methodology introduces two systematic documentation structures, an Analysis Library and a Design Library. The Analysis Library records findings about the business and its needs for the information system in question, and is created in terms understandable to users in the business. The Design Library records the logical inventions and solutions the developers have discovered which address the needs of the business in a potential information system, and uses more a precise and concise design language understood by the developers. Both libraries are organized in the same way, in the box structure of the information system under development. The developers understand and create both libraries,

using the Analysis Library to interface with management and users in the language of the business, and the Design Library to ensure the completeness and consistency of the information system in more technical language.

For example, the results of management or user interviews will appear first in the Analysis Library, and be confirmed in that form with the management or users. Such interviews may not cover unusual cases in computer operations that the users never see, such as how files are protected during an electrical power outage. But as those results are translated into the Design Library, additional technical considerations may arise, as in this case, how power outages are to be handled. Once problems and solutions are recorded in the Design Library, their results may be fed back to the Analysis Library and subsequent discussion with management or users, in this case to decide whether to provide for emergency power facilities for computer operations.

Box structures provide continuity of form for managing system development through its life cycle. They can be used extensively and similarly in the three principal phases of development, namely explanation, specification, and implementation:

Explanation: Do the developers understand the problem? They can demonstrate they do by describing current operations, manual or automated, with a comprehensive treatment of inputs, outputs, storage and processing. Formulated in box structures, these descriptions should be verified with the users and management.

Specification: Do the developers have a solution to the problem? They can demonstrate they do by describing a possible information system to improve current operations, with a comprehensive treatment of the inputs, outputs, storage and processing proposed. Formulated in box structures, the proposal information system should be augmented with cost and schedule estimates for management consideration.

Implementation: Can the developers make good on their proposed solution? The box structured specification is the right foundation for a box structured implementation in a top down hierarchical development of the information system to meet specifications within budget and schedules.

All the above is easier said than done, but a common box structured methodology that provides intellectual continuity through all three phases makes the management significantly more effective.

4.3 Managing Box Structured Development

Hierarchical box structures provide a natural framework for cost and schedule control. Once analysis is completed, the initial design task is to develop a top level black box, state machine, and clear box. The clear box will make use of black boxes at the next level of refinement, for which the design process will repeat. The top level design effectively partitions the original design problem into a structure of component problems, each of which can be dealt with independently using the same box structure methodology. Each new black box in the structure represents a new top, which must in turn be elaborated into a box structure hierarchy of its own. Since each new black box is smaller and simpler than those above it in the hierarchy, eventually black boxes will be reached which do not introduce new black boxes, and the design will be complete.

The principal benefit of box structure design lies in new opportunities for management planning and control of information system development. Box structures permit a rigorous design to cost management process, in the stepwise allocation and consumption of project resources. Every new black box in the evolving hierarchy represents a new subproject to be designed to cost given the resources available.

REFERENCES

- [Baker] F. T. Baker, "Chief Programmer Team Management of Production Programming," *IBM Systems Journal*, Vol. 2, No. 1, 1972, pp. 56-73.
- [Baker] F. T. Baker, "System Quality Through Structured Programming," 1972 Fall Joint Computer Conference, Vol. 41, Part I AFIPS, 1972, pp. 339-343.

- [Linger] R. C. Linger, H. D. Mills, and B. I. Witt, Structured Programming: Theory and Practice, Addison Wesley, 1979.
- [Madden] Madden, W. A., and Rone, K. Y., "Design, Development, Integration: Space Shuttle Primary Flight Software System," Communications of the ACM, Vol. 27, No. 9, Sept. 1984, pp. 914-925.
- [Mills] A. Mills, "Smoothing in Inventory Processes" in Essays in Mathematical Economics, M. Shubik (ed.), Princeton University Press, 1976, pp. 131-148.
- [Yourdon] Yourdon, E. (ed.), Writings of the Revolution: Selected Readings on Software Engineering. Yourdon Press, New York, N.Y., 1982.

**DPMA GRADUATE MODEL CURRICULUM IN MANAGEMENT INFORMATION SYSTEMS
A PRELIMINARY REPORT**

Doris G. Duncan, PhD, CDP, CSP, CMI
California State University, Hayward

ABSTRACT

The purpose of the graduate model curriculum in MIS is to provide a guideline to colleges and universities interested in establishing (or revising) either a general Master of Business Administration (MBA) program with an area of emphasis in MIS or an in-depth Master of Science (MS) program in MIS. This curriculum acknowledges the growing importance of information as a resource, much as land, labor and capital. It provides the framework for effective analysis, design, implementation and management of complex computer-based information systems.

As the need increases for faster, more effective decision making within complex organizations, so does accurate and timely access to information become more critical. The Management Information Systems (MIS) curriculum outlined here acknowledges the growing importance of information as a resource, much as land, labor and capital. It provides the tools to effectively analyze, design, implement and manage complex computer-based information systems.

THE IMPETUS

The DPMA Model Curriculum for Undergraduate Computer Information Systems Education was well-received and is now being updated. Development of a DPMA graduate model curriculum in computer-based information systems is an appropriate follow-on project. In order to emphasize the decision making orientation of the graduate model curriculum, the discipline is called "Management Information Systems" rather than "Computer Information Systems."

Both students and employers should benefit from this graduate model curriculum. It is intended to prepare students to participate effectively in areas of business impacted by fast changing technologies by developing both their management insights and technical skills for analysis and maintenance of management information systems. Appropriate starting positions for graduates include information systems analyst, systems designer, data base administrator, consultant, and marketing support. Eventually graduates may progress through such positions as project leader, MIS manager and chief information officer. The graduate model curriculum is not intended to prepare graduates for programming positions; this is left to the undergraduate model curriculum. Hopefully employers will benefit from increased professionalism and a standard for recruiting for specific positions as well as from availability of better qualified graduates.

OBJECTIVES

Objectives developed so far for the graduate model curriculum focus on the overall purpose of the curriculum and desired outcome for graduates:

(1) A national educational standard for the discipline of Management Information Systems (MIS) at the graduate level will be developed. The model curriculum should be useful to colleges and universities developing or revising either an MBA program with an emphasis in MIS or an in-depth MS program in MIS.

(2) The graduate model will complement the DPMA Model Curriculum for Undergraduate Computer Information Systems.

(3) The model curriculum will be endorsed and supported actively by DPMA.

(4) The primary users will be colleges and universities which offer masters degrees in business administration.

(5) Graduates educated under this curriculum should be capable of performing a broad range of professional and managerial computer and information systems assignments.

(6) Graduates should be familiar with business functions and operations and understand the role of MIS in business.

(7) Graduates should be able to communicate ideas and project results orally and in written form.

(8) Graduates should be able to work independently and as members of a team.

(9) Graduates should understand and be able to apply a formal process for decision making and problem solving.

(10) Graduates should be able to study and solve information problems within organizations.

(11) Graduates should be capable of implementing project management principles.

(12) The MIS professional should leave school having learned how to learn and be committed to a life-long program of continuing professional education.

Selection of faculty qualified and interested in implementing the model curriculum is paramount to achieving these objectives.

ACCREDITATION CONSIDERATIONS

Due to core and breadth requirements, schools accredited by the American Assembly of Collegiate Schools of Business (AACSB) are somewhat limited in the number of courses they can require in any specific speciality within a one-year MBA program. However, they should have ample space in two-year programs to incorporate all ten courses outlined in the MIS program. Many schools may wish to establish both an MBA specialization requiring four to five of the graduate MIS courses and also an in-depth MS program which requires all ten courses.

Schools not accredited by the AACSB, nor interested in seeking AACSB accreditation, have more flexibility in their Masters degree programs. They should find it relatively easy to implement all ten courses, and perhaps one more considered suitably related by the administering department.

Doctor of Philosophy programs in MIS are generally very unique. Nevertheless, some schools may choose to make certain courses in the MIS Masters program part of the PhD course work. They may also choose to add a few courses such as "Seminar in Advanced MIS Topics" or "Seminar in Current MIS Research," a forum for PhD candidates to present current research on their dissertations.

PREREQUISITES

Students should complete undergraduate business core requirements prior to matriculation into the graduate MIS program. Core requirements for business majors normally required for AACSB accreditation include:

- (1) Financial Accounting Principles
- (2) Managerial Accounting Principles
- (3) Quantitative Methods
- (4) Principles of Management
- (5) Principles of Marketing
- (6) Principles of Finance
- (7) Organizational Behavior
- (8) Production and Operations Management

It is important also that each student have sufficient general business background and the communications skills necessary for comfortable interaction with computer system users and all levels of management.

Prior to matriculation into the MIS program, it is highly desirable (1) that students be familiar with systems and program development methodologies, (2) that they are acquainted with the systems development life cycle, and (3) that they be fluent in at least one programming language. It is specifically recommended that prior to enrolling in graduate MIS program students complete the undergraduate CIS courses shown below:

CIS/86-101 Introduction to Computer Information Systems

Emphasis is on computer requirements, history, hardware functions, programming, systems development and computer operations. Hands on exercises in programming and business applications are recommended.

CIS/86-2 Introduction to Business Application Programming

This is an introductory course in program design and development. Students work to a structured development process which involves five phases for understanding, design, specification, coding and checking programs. Pseudocode development will not be language specific, however programs are written using COBOL.

CIS/86-4 Data Files and Data Bases

This course stresses the prevalence of data bases in the world of computing. It includes application development through fourth and fifth-generation programming techniques and fundamentals of data structures, normalization of data, data modeling and construction of data base schema.

CIS/86-5 Systems Development Methodologies: A Survey

This course deals with traditional analysis, design, and implementation through the data flow analysis and systems development life cycle approach. It includes data structures, data definition and normalization.

Course numbers and descriptions of prerequisites are based on working documents distributed by DPMA in connection with revising the undergraduate model curriculum (4).

THE DPMA GRADUATE MODEL CURRICULUM IN MIS

Following is a summary of specific courses recommended for Masters degree programs in MIS. MIS/86-1 and MIS/86-2 are the only required courses in the graduate program and are prerequisites to the elective courses. It is recommended that students take MIS/86-3 through MIS/86-10 in sequence wherever possible in order to most effectively integrate new concepts with those previously introduced.

Required MIS Courses:

MIS/86-1 SEMINAR IN INFORMATION ANALYSIS

Comparative theory and practice of MIS. Topics include systems life cycle, requirements definition of user needs, feasibility studies, alternative solutions, software selection and development, testing, implementation, documentation and conversion of computer-based information systems. User-system interfaces, user interviewing techniques, standard systems analysis tools as data flow diagrams, structure charts and prototyping will be explored. The information systems analyst as consultant.

MIS/86-2 SEMINAR IN DECISION SUPPORT SYSTEMS

Conceptual foundation in evaluation and application of information systems tools in problem solving and making decisions. Topics include formulation, development and evaluation of decision making models, simulation techniques, optimization, expert systems, artificial intelligence. Hands-on use of decision support tools such as data bases internal and external to the organization.

Elective MIS Courses:

MIS/86-3 SEMINAR IN DISTRIBUTED PROCESSING AND TELECOMMUNICATIONS

Overview of geographically distributed computer-communications facilities. Network design, structure and optimization are addressed. Regulated common carriers, data transmission, routing techniques, reliability, protocols, error detection, response time, polling and contention, teleconferencing, communication media as terminal devices, modems, and controllers are included.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-4 SEMINAR IN DATA BASE MANAGEMENT AND ADMINISTRATION

Management of data as a resource. Development of a conceptual framework to evaluate, select, acquire, install and maintain commercial data base management packages for use in MIS. Data structures, data storage, data representation, data flow, data dictionaries, expert systems, and programming languages are explored. Concurrency and configuration control, access methods and utilities are included.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-5 SEMINAR IN AUDIT, CONTROL AND LEGAL ASPECTS OF MIS

Exposure to audit techniques, types of controls, data integrity, risk assessment and professional standards in MIS auditing. Computer abuse, ethics, privacy, and security are emphasized. Legal considerations include contracts, copyrights, patents, trade secrets, warranties, product liability, and the role of regulatory agencies in MIS. Case studies will be used.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-6 SEMINAR IN OFFICE SYSTEMS

Text-based office systems, including word processing, graphics, electronic mail, facsimile, electronic filing and retrieving, records management, calendar management, video conferencing. Integration with data-based processing systems, and advanced input and output techniques such as the mouse and laser printers. Human-machine interfaces, technological trends in data processing and the office environment.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-7 SEMINAR IN PROJECT MANAGEMENT

The organizing, staffing, budgeting, scheduling, and coordinating of MIS projects. Emphasis is on programming and systems development teams, programming productivity aids, management tools, milestone requirements, documentation of maintained systems, post-implementation followup and user participation. Performance measurement, use of outside services and organization of facilities (centralized vs. decentralized vs. distributed) included. A term project which requires application of project management will be assigned.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-8 SEMINAR IN INFORMATION RESOURCE MANAGEMENT

Organization planning and controlling of user services and the corporate information center. Information as a corporate asset. Integration of personal computers with mainframes. Convergent technologies as voice, data and video transmission, video conferencing, local area networks, and work stations. Impact of electronic technology on traditional information systems processing organizations, management methodologies and career opportunities.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-9 SEMINAR IN MIS PLANNING

Application of the planning cycle: operational, tactical and strategic. Considerations of hardware and software technology trends, system migration and compatibility, capacity planning, contingency planning, service level management, project selection and prioritization. Long-range staffing with emphasis on use of contract programmers and consultants to supplement regular staff. Comparison of internal systems development with systems available for purchase. Impact of new technologies and professional societies and certification.

Prerequisites: MIS/86-1 and MIS/86-2

MIS/86-10 CAPSTONE RESEARCH PROJECT IN MIS

Development and writing of a final research paper under supervision of an advisor. Oral defense required. Paper may be a) a research report on a current MIS topic, b) a summary of a hypothetical case or c) a summary of an MIS project developed for actual experience.

Prerequisites: MIS/86-1 and MIS/86-2

Finally, if space in their curriculum permits, some colleges may wish to add MIS/86-11, which would be any graduate-level course deemed related to MIS and approved by the appropriate department.

Suggestions regarding this graduate model curriculum in MIS are encouraged. Please direct your comments to:

Professor D. G. Duncan
Information Systems Management
School of Business and Economics
California State University, Hayward
Hayward, California 94542
(415) 881-3595

SELECTED REFERENCES:

- (1) Adams, David R. and Thomas H. Athey, eds., DPMA Model Curriculum for Undergraduate Computer Information Systems Education, Park Ridge, IL: DPMA Education Foundation, 1981.
- (2) Duncan, Doris Gottschalk, "Great Expectations for Information Systems Curriculum: A Survey of MIS Management," Interface, The Computer Education Quarterly, Summer, 1984, pp. 14-17.
- (3) Nunamaker, Jay F. Jr., J. Daniel Couger and Gordon B. Davis, eds., "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs," Communication of the ACM, November, 1982, pp. 781-805.
- (4) Price, Don, ed., "CIS Curriculum '86: Evaluation Survey and Working Document," DPMA Education Foundation, 1985.

ASSOCIATE DEGREE MODEL CURRICULUM
for: Computer Information Systems

Rod B. Southworth
Laramie County Community College

The Associate Degree Model Curriculum for Computer Information Systems, hereafter referred to as simply the "Model Curriculum," is the result of efforts initiated and supported by DPMA. DPMA has been actively involved with curriculum development since 1981, when it first published a four-year CIS model curriculum which has subsequently been adopted by hundreds of colleges and universities. The 1981 model curriculum is currently being upgraded with a significant number of improvements reflecting the dynamic nature of the subject matter. DPMA initiated a model curriculum for the secondary schools and completes the series with this associate degree model.

The Model Curriculum will serve as a guideline for development of two-year degree programs designed to meet identified learning needs for entry-level computer programming and operations personnel, as well as for the developing new skills to meet requirements of computer users. The Model Curriculum was developed as a joint effort by an Advisory Committee composed of three groups of participants:

- (1) Eight college instructors in the computer field, representing both small and large two-year colleges.
- (2) Media representatives from Information/Education, Inc., a company that develops textbook and other educational materials, including a series of textbooks for the four-year CIS model curriculum.
- (3) The DPMA Curriculum Activities Manager, Don Price.

The Advisory Committee recognized that there were special needs associated with the development of curricula for two-year colleges. Because the major emphasis of most two-year programs is on applied learning, the Model Curriculum represents a blending of computer, business, and communication elements. It is not to be confused with four-year Baccalaureate programs in Computer Science or Computer Engineering, which are considered to be more "academic," rather than "applied," in nature.

The Advisory Committee undertook to develop a series of CIS courses that could be selectively used by two-year colleges to meet the wide variety of educational requirements to prepare students for entry-level positions in the computer field. It was anticipated that general education courses would need to be included, in order to meet differing degree requirements and student needs. Students not desirous of a degree could also benefit by selecting those courses required to upgrade their vocational skills. It is recommended that students desiring a four-year degree select only those courses from the two-year program that will transfer to a four-year program, since transferability is not the primary purpose of the Model Curriculum.

There are four (4) separate learning sequences, or tracks, in the Model Curriculum. These tracks were developed to correspond to the current labor market for computer personnel. Additionally, these four tracks may be organized into two major groups as follows:

Group I -- Computer Information Systems

- Track 1. Using Microcomputers
- Track 2. Programming

Group II -- Computer Operations

- Track 3. Computer Operating
- Track 4. Data Entry/Transaction Processing

Group I tracks are directed specifically toward the practical applications of computers for problem solving and operational business situations. Satisfactory completion would qualify students for jobs requiring intensive interactive use of microcomputers or for entry-level positions as programmers.

Group II tracks will develop skills that would qualify students for entry level employment in the occupations of computer operator, peripheral operator, tape librarian, data control clerk, or data entry operator. The student could also qualify for the relatively new position of microcomputer bookkeeper, and be responsible for transaction and/or document processing.

For each of these tracks, the curriculum document outlines the recommended core and elective course offerings. The corresponding detailed course descriptions represent a consensus of requirements identified by educators and practitioners in the CIS field. The suggested instructional guidelines provide instructors the necessary freedom to tailor presentations to the specific needs of the student.

A copy of the Model Curriculum is available to those attending this session. Additional copies can be obtained from:

Data Processing Management Association
Attn: Hildegard Klemm
Mgr. of Educational Services
505 Busse Highway
Park Ridge, IL 60068-3191

Rather than review this extensive document in detail, I would urge you to study it carefully, using it wherever applicable, to assist you in developing an up-to-date curriculum that best meets the needs of your students. You may also find this document to be very helpful in "selling" curriculum changes to your administration.

I would like to demonstrate how this document can be used to structure a two-year degree program by using my community college as an example. We have an Associate Degree program for CIS majors which is designed to prepare students for entry-level programming positions. In addition to the traditional CIS content, this track recognizes the need for students to understand the application of microcomputers, database principles, and high-order software. Throughout the curriculum, major emphasis is placed on the application of the computer as a tool for solving business-related problems. Our program follows the Model Curriculum closely, even though minor modifications were required to meet our institutional requirements. For purposes of clarity, course numbers and titles defined in the Model Curriculum document have been used in the following example.

Associate of Applied Science Degree in CIS

Laramie County Community College
Cheyenne, Wyoming

Computer Courses (24 credit hours):

Introduction to Computer Applications - CIS 1
Programming Design and Development - CIS 2
Computer Systems Development - CIS 3
Programming Language I (RPG) - CIS 4
Programming Language I (COBOL) - CIS 4
Programming Language II (Adv. COBOL) - CIS 5
Systems Analysis and Design - CIS 16
Microcomputers in Business - CIS 6 (elective at
LCCC)
Microcomputer Operating System Concepts - CIS 7
(elective)
Microcomputer Packages - CIS 15 (elective)

Business Courses (12 credit hours):

Introduction to Business
Principles of Management
Accounting Principles I
Accounting Principles II

General Education Courses (29 credit hours):

English Composition
Business Communications
College Algebra
Math Analysis
Logic
Science
U.S. and Wyoming Government
Organizational Human Relations
Elective (open)

COBOL 8X - Teaching the New Compiler

Robert T. Grauer, Ph. D.
University of Miami, Coral Gables, Florida

ABSTRACT

This paper highlights changes in the 8X compiler, with emphasis on those modifications designed to improve structured programming in COBOL. It discusses the evolution of COBOL, explaining the delay in the acceptance of the new standard. The paper concludes that, despite the attractiveness of new features in COBOL 8X, COBOL 74 should continue to be stressed in the classroom due largely to the maintenance burden which exists in industry.

EVOLUTION OF COBOL

At its inception in 1959, COBOL was designed as "open ended and capable of accepting change and amendment." The revision process is the province of the COBOL committee of CODASYL (Conference on DATA SYSTEMS Languages), which meets periodically to publish a JOD (Journal of Development). This document is submitted to the X3J4 Technical Committee of the American National Standards Institute (ANSI), which publishes the actual standard. Until recently there have been two official standards, COBOL 68 and COBOL 74, the last version known officially as American National Standard COBOL X3.23-1974.

In 1977 the X3J4 committee began revising the 1974 standard based on the CODASYL JOD of 1976. Four intermediate publications, COBOL Information Bulletins 17, 18, 19, and 20, provided the public with its first glimpse of what to expect. CODASYL subsequently published a JOD in 1978 which revised the 1976 publication.

In June 1981 the X3J4 committee completed its work and approved the content of a draft proposal to revise ANS COBOL X3.231974. The draft was made available in September 1981 for public comment, with the review period ending in February 1982. The proposed standard was again revised to reflect public reaction, and made available for review in the fall of 1984. Final approval is expected shortly.

OBJECTIONS TO COBOL 8X

The original goal of the committee was to have the new standard available in 1980, maintaining a six year cycle between standards. Adverse public reaction, however, continually postponed the approval date. After waiting for, but not receiving COBOL 80, 81, and 82, many people have taken to calling the new standard COBOL 8X.

Most objections were over incompatibility with the existing compiler. Introduction of new reserved words, for example, posed compatibility problems if existing programs inadvertently used a (new) reserved word as a data name. Other incompatibilities resulted from the deletion, clarification, or reinterpretation of existing features.

Another problem in the evolution of the new standard was that the CODASYL committee became carried away with esoteric, often downright silly, changes. One 1981 proposal (which has been removed from the 1984 revision), suggested that tables of up to 48 levels be permitted. To appreciate the sheer absurdity of this extension, consider that a programmer actually does create such a table, and, further, that each of the 48 OCCURS clauses has the minimum of two elements. One is now confronted with the staggering number of 2^{48} (or approximately 10^{15}) elements.

Assuming a CPU speed of 1,000,000 operations per second, merely initializing the table would require one billion seconds or approximately 33 years of CPU time. (This does not include the time necessary to page in and out from disk).

Storing a table of 10^{15} elements presents additional difficulty. If each data element required only a single byte of storage, and if a disk drive with a capacity of one billion bytes were available, then one million disk packs would be required to store the entire table. The point is simply that the proposed extension to a 48 level table, (and other similar suggestions) were utterly foolish, and needlessly lengthened the approval cycle.

AVAILABILITY OF THE NEW COMPILER

Despite the difficulties in obtaining official approval for a new standard, many vendors have made the essential features of the 8X compiler available to their customer base. In early 1984 IBM announced OS VS COBOL II (supporting some, but not all of COBOL 8X) for its installed base of mainframe customers.

IBM's tacit approval makes it safe to assume that improvements in the new compiler will work their way into common use over the next several years. The changes will be slow in coming, (recall the conversion problems from 68 to 74), but they will eventually be realized. The remainder of this paper highlights the most essential features.

STRUCTURED PROGRAMMING ENHANCEMENTS

By far, the biggest improvement in COBOL 8X over its predecessor is its accommodation of structured programming concepts. The new standard is the first to be developed after the common acceptance of the structured methodology. Hence, an underlying objective of the standards committee was to make COBOL more conducive to current programming theory. Accordingly, COBOL 8X makes specific provision for:

- (1) The DO WHILE and DO UNTIL constructs through modification of the PERFORM statement with a TEST BEFORE or TEST AFTER facility.
- (2) Improved readability through introduction of scope terminators, such as ENDIF, which make it possible to nest conditional statements.
- (3) The case construct via introduction of the EVALUATE statement.
- (4) False condition branch (e.g. NOT AT END), which provides structured symmetry to conditional clauses.

Each of these features is discussed in detail.

PERFORM Statement

The UNTIL format of the PERFORM verb has been expanded to allow TEST AFTER and/or TEST BEFORE, with the latter as default:

$$\text{PERFORM} \left[\text{procedure-name-1} \left[\left\{ \begin{array}{c} \text{THROUGH} \\ \text{THRU} \end{array} \right\} \text{procedure-name-2} \right] \right]$$
$$\left[\text{WITH TEST} \left\{ \begin{array}{c} \text{BEFORE} \\ \text{AFTER} \end{array} \right\} \right] \text{UNTIL condition-1}$$
$$\left[\text{imperative-statement-1} \text{ END-PERFORM} \right]$$

In other words, COBOL now accommodates both a DO WHILE and a DO UNTIL. The TEST BEFORE condition tests before performing the procedure and corresponds to a DO WHILE; i.e. if the condition is satisfied initially, the designated procedure is never executed. TEST AFTER, on the other hand, corresponds to the DO UNTIL construct; consequently, if the condition is satisfied initially, the designated procedure will still be executed once.

The COBOL 8X syntax shows the procedure name as an optional entry, enclosed in square brackets. This in turn provides an in-line capability for the PERFORM statement; e.g.:

```

PERFORM
  statement-1
  statement-2
  .
  .
  .
  statement-n
END-PERFORM

```

An inline PERFORM functions according to the general rules of a regular PERFORM statement, except that the statements executed are those contained within the PERFORM statement itself, i.e., between PERFORM and END-PERFORM. (Accordingly, omission of procedure-name-1 requires that both imperative-statement-1 and END PERFORM be coded.) The inline PERFORM does not provide any additional logic capability. Nevertheless, it is a welcome addition in that it can eliminate a degree of page turning and, further, gives COBOL a similar capability to that found in other block oriented languages such as PASCAL and PL/I.

IF STATEMENT

The power and readability of the IF statement has been tremendously improved through inclusion of the END-IF scope terminator. An abbreviated format of the IF statement in COBOL 8X is

```

IF condition-1 THEN {statement-1} . . .
{ ELSE {statement-2} . . . END-IF }
END-IF

```

Although an IF statement may be terminated by either an ending period or an END-IF scope terminator, it is strongly recommended that END-IF terminators be used exclusively. Inclusion of END-IF will neatly eliminate the "column 73" problem that existed in COBOL 74. Consider Figure 1, which contains apparently straight forward COBOL code and its surprising output.

The logic in Figure 1 is straight forward, yet the output is unexpected. An order of \$2,000 or more receives a discount of 2% on the entire order. The amount due (NET) is equal to the amount ordered minus the discount. The code seems correct, yet the calculated net amounts are wrong for any order less than 2,000. The net amount printed for these orders equals the net for the previous order (i.e., the net for an order of 1,000 is incorrectly printed as 3,920, which was the correct net for the preceding order of 4,000). The net amount for an order of 1,500 was printed as 4,900, and so on. Why?

COBOL code:

```

IF AMOUNT-ORDERED-THISWEEK < 2000
  MOVE ZEROS TO CUSTOMER-DISCOUNT
ELSE
  COMPUTE CUSTOMER-DISCOUNT = AMOUNT-ORDERED-THISWEEK * .02
  COMPUTE NET = AMOUNT-ORDERED-THISWEEK - CUSTOMER-DISCOUNT
  DISPLAY AMOUNT-ORDERED-THISWEEK
  CUSTOMER-DISCOUNT NET

```

Period terminating compute is in column 73

Inadvertently taken as part of ELSE clause

Output:

Amount Ordered	Discount	Net
3000	60	2940
4000	80	3920
1000	0	3920
5000	100	4900
1500	0	4900

Correct calculation

Net is incorrect and equal to value of previous order

Figure 1 - The Missing Period

The only possible explanation is that the COMPUTE NET statement is not executed for net amounts less than 2,000. The only way that can happen is if the COMPUTE NET statement is taken as part of the ELSE clause, and that can happen only if the ELSE is not terminated by a period. The period is present, however, so we are back at ground zero - or are we? The period is present, but in column 73, which is ignored by the compiler. Hence the visual code does not match the compiler interpretation, and the resulting output is incorrect.

A period in column 73 is not a contrived problem. Inclusion of the END-IF delimiter will eliminate future errors of this type, and the IF statement should be recoded as follows:

```

IF AMOUNT-ORDERED-THISWEEK < 2000
  MOVE ZEROS TO CUSTOMER-DISCOUNT
ELSE
  COMPUTE CUSTOMER-DISCOUNT = AMOUNT-ORDERED-THISWEEK * .02
END-IF.

```

The END-IF scope terminator also provides additional logic capability by permitting the nesting of conditional statements. Consider Figure 2 and its implementation in COBOL 74 and COBOL 8X.

The END-IF terminator effectively transforms a conditional statement to an imperative (i.e., complete) statement. Hence the required logic of Figure 2a can be expressed as a single IF statement in COBOL 8X. By contrast, the COBOL 74 implementation requires an additional PERFORM statement. Scope terminators are available for a host of other verbs (e.g., READ AT END, COMPUTE SIZE ERROR, etc.).

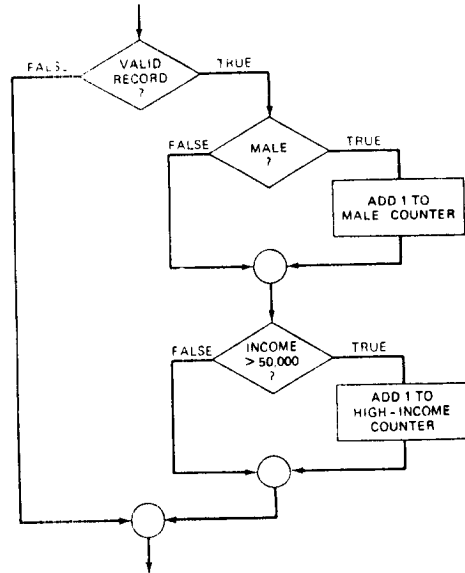


Figure 2a - Required Logic

```

IF VALID-RECORD-SW = 'Y'
  IF SEX = 'M'
    ADD 1 TO MALE-COUNTER
  END-IF
  IF INCOME > 50000
    ADD 1 TO HIGH-INCOME-COUNTER
  END-IF
END-IF

```

Figure 2b - COBOL 8X Implementation

```

IF VALID-RECORD-SW = 'Y'
  PERFORM DO-ADDITIONAL-TESTS.
.
DO-ADDITIONAL-TESTS.
  IF SEX = 'M'
    ADD 1 TO MALE-COUNTER.
  IF INCOME > 50000
    ADD 1 TO HIGH-INCOME-COUNTER.

```

Figure 2c - COBOL 74 Implementation

EVALUATE

COBOL 74 was limited in its ability to implement the case construct directly, as the programmer was forced to use either a multilevel nested IF or a GO TO DEPENDING statement. Although both methods work, neither is ideal and the resulting code was often obscure.

COBOL 8X remedies the situation through introduction of the EVALUATE statement, whose abbreviated format is shown:

```

EVALUATE {identifier-1}
          {expression-1}

WHEN {condition-1}
     {TRUE
      FALSE} imperative-statement-1

[WHEN OTHER imperative-statement-2]

END-EVALUATE

```

REFERENCES

- 1) Structured COBOL Programming, Robert T. Grauer, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1985.
- 2) Draft Proposed Revised X3.23 American National Standard Programming Language COBOL, September 1983, Technical Committee X3J4, American National Standards Institute, New York, N.Y.

Consider, for example, a program in which processing depends on the value of an incoming "year in school" code. When the code is equal to 1, a freshman routine is to be performed; when the code is equal to 2, a sophomore routine, and so on. The multi-branch situation is easily implemented as follows:

```

EVALUATE YEAR-IN-SCHOOL
  WHEN 1 PERFORM FRESHMAN
  WHEN 2 PERFORM SOPHOMORE
  WHEN 3 PERFORM JUNIOR
  WHEN 4 PERFORM SENIOR
  WHEN OTHER PERFORM PROCESS-OTHER
END-EVALUATE.

```

In the example YEAR-IN-SCHOOL is evaluated. If it is equal to 1, FRESHMAN is performed; if it is equal to 2, SOPHOMORE is performed, and so on. Observe the use of the reserved word OTHER to accommodate an error processing routine, and the terminating entry, END-EVALUATE.

FALSE CONDITION BRANCH

The syntax for the READ statement in the new compiler is as follows:

```

READ file-name

[AT END imperative-statement-1]

[NOT AT END imperative-statement-2]

[END-READ]

```

The addition of the NOT AT END clause provides a symmetry for the conditional branch that was not present in COBOL 74. Taken together with the new inline PERFORM, these two statements neatly eliminate the need for a priming READ. Hence the "mainline" portion of a COBOL 8X program will now contain the following:

```

PERFORM UNTIL DATA-REMAINS-SWITCH = 'NO'
  READ INPUT-FILE
  AT END
    MOVE 'NO' TO DATA-REMAINS-SWITCH
  NOT AT END
    PERFORM PROCESS-RECORD
END-READ
END-PERFORM.

```

The combination of the inline PERFORM together with the false condition branch of the READ statement completely alters the appearance of the driving paragraph in a COBOL program. The priming READ statement can be eliminated, and the resulting program truly reads in top down fashion.

CONCLUSION - COBOL 74 VERSUS COBOL 8X

Despite the considerable attractiveness of the new compiler, the author advocates that COBOL 74 continue to be stressed in the classroom. COBOL 8X may still be covered, but the 74 syntax, and its peculiarities (e.g. the priming READ) should be taught first. A logical division would be to cover the old compiler exclusively in the first semester, and introduce COBOL 8X in the advanced course.

This rather pessimistic view recognizes the reality that COBOL 74 will be around commercially for some time (if not forever) due to the overwhelming maintenance burden which exists today. Instructors would be doing their students a tremendous disservice to disregard COBOL 74, when that is the compiler that their graduates will most likely encounter. It took industry many years to convert from COBOL 68 to COBOL 74, and history has a tendency to repeat itself.

AN EVALUATION OF INFORMATION SYSTEMS
TEACHING METHODOLOGIES:
SELF-STUDY VS. LECTURE

GEORGE W. MORGAN, Southwest Texas State University

R. WAYNE HEADRICK, Texas A&M University

WALTER E. JOHNSTON, Southwest Texas State University

To provide adequate coverage of a course newly required of all Business majors with the resources available at the time, a self-study teaching format was adopted. Subsequently, an experiment was designed and undertaken to determine the impact of the self-study format on the scholastic performance of the students. The results of the experiment indicate that the level of a self-study student's prior knowledge of computers directly affects his/her scholastic performance level, while no such relationship was indicated for the student in the control, or traditional lecture, environment.

INTRODUCTION

In the Fall semester of 1982, Introduction to Data Processing was added to the common body of knowledge required of all Bachelor of Business Administration degree candidates of Southwest Texas State University (SWTSU). Consequently, over one thousand students began requesting that course each semester. As the Department of Accounting and Computer Information Systems could facilitate only three hundred students in a traditional classroom setting, it was determined that an alternative teaching methodology would be required to enable the Department to provide adequate course coverage with the available resources.

A Self-Study methodology for teaching the Introduction to Data Processing course was selected for implementation in the Fall semester of 1982. The objectives of the course were to provide students with knowledge of the basic elements of computer literacy as defined by the Texas Education Agency (i.e., computer related terminology and the use of computers in society, history and development of computers, use of the computer as a tool, communicating instructions to the computer, and careers in the computer field), and to provide an introduction to the analysis and design of business information systems as required in the American Assembly of Collegiate Schools of Business (AACSB) curriculum.

Students enrolled in the Self-Study course were not required to attend any regularly scheduled classes. Several learning resources were, however, made available to any student who wished to use them. Those resources included:

- (1) A textbook selected for its readability
- (2) A problems/questions workbook that accompanied the textbook
- (3) A professionally produced film series coordinated with the textbook
- (4) A regularly scheduled tutorial lab staffed by senior level Computer Information Systems majors
- (5) Regularly scheduled optional lecture sessions

Examination dates and topics included were provided to the students when they registered for the course. The examinations consisted of objective, multiple-choice and true/false questions selected from a test bank provided by the author of the textbook, and generated

based on the contents of the film series and student workbook.

Since this Self-Study methodology represented a significant departure from the classroom lecture teaching methods normally used in the SWTSU School of Business. A formal study was conducted during the Summer I semester of 1983 to evaluate the impact of the Self-Study format on the students. The research methodology used and results of that study are as follows.

METHODOLOGY

To best judge the relative impact on scholastic performance of the Self-Study methodology, the students enrolled in the Introduction to Data Processing course were randomly placed in either a Self-Study or a Control section in a 4-to-1 ratio, until the Control section was full. The Control section was taught using a traditional lecture setting, covering the same material as that found in the learning resources available to the students enrolled in Self-Study sections. To provide for subsequent in-depth analyses, appropriate demographic information about the students in both the Self-Study and Control groups was gathered through the use of a "student verification of enrollment" form. As the students were not allowed to take the first examination until they completed the form, a one hundred percent return rate was achieved.

The scholastic performance of both groups was examined by comparing the results of a pre-test, given at the beginning of the semester to assess the students' level of prior knowledge of computers, with the final semester averages. To determine whether the teaching methodology has a direct effect on the students' scholastic performance, the covariance analysis technique was employed.

Covariance analysis^{1,2,3} provides the capability to assess the contribution of one uncontrolled independent variable (pre-test score) and two classes (teaching methodology), to the dependent variable (final course average). Specifically, covariance analysis provides for correcting statistically for the effects of prior knowledge that could not be properly standardized between the two teaching methodologies.

Covariance analysis can be used to

- (1) test for homogeneity in slopes between classes,

(2) test for homogeneity in intercepts between classes (with slopes assumed constant for all classes) and

(3) test for homogeneity in the complete relationship between classes,

with the tests being performed in the order indicated^{3,4} until the null hypothesis associated with one of the tests is rejected. Upon rejection of a null hypothesis, it is inappropriate to continue with any subsequent test³.

Considering the sample data to have p classes, n total students (n_i students in class i) and $k-1$ uncontrolled variables, the complete analysis of covariance can be summarized as indicated in Table 1.

Test	Null Hypothesis	F-test	degrees of freedom
1	homogeneous slopes	$\frac{\text{mean square (differential slopes \& intercepts)}}{\text{mean square (residual)}}$	$\frac{pk - p - k + 1}{n - pk}$
2	homogeneous intercepts	$\frac{\text{mean square (differential intercepts)}}{\text{mean square (residual)}}$	$\frac{p - 1}{n - p - k + 1}$
3	overall homogeneity	$\frac{\text{mean square}}{\text{mean square (residual)}}$	$\frac{1}{n - k}$

Table 1: Analysis of Covariance

RESULTS

As indicated above, the study consisted of dividing 145 Introduction to Data Processing students ($n = 145$) into Self-Study and Control groups ($p = 2$), with the pre-test being the only uncontrolled variable ($k-1 = 1$ and the final course average being the dependent variable. As the various sums of squares and mean squares are readily obtainable, they were computed prior to the accomplishment of any tests of hypothesis. The results of those computations are provided in Table 2. Since the null hypothesis that

Test	Sums of Squares	d.f.	Mean Squares	F	significance
1	SS_R 429.6	1	429.6	6.44	$\alpha < .01$
	SS_E 9408.3	141	66.7		
2	SS_R 319.1	1	319.1	4.61	n.a.
	SS_E 9837.9	142	69.3		
3	SS_R 901.4	1	901.4	12.69	n.a.
	SS_E 10157.0	143	71.0		

Table 2: Analysis of Covariance Computations

there is homogeneity in slopes between groups must be rejected, further testing is inappropriate. The equations that minimize the residual sums of squares when relating final course averages (Y) to pre-test scores (X) are

$$Y = 60.1 + 0.50X \text{ for the Self-Study group}$$

$$Y = 71.2 + 0.00X \text{ for the Control group.}$$

As illustrated by Figure 1, the resultant equations indicate no relationship between final course averages and pre-test scores within the Control group, while a direct relationship is indicated within the Self-Study group.

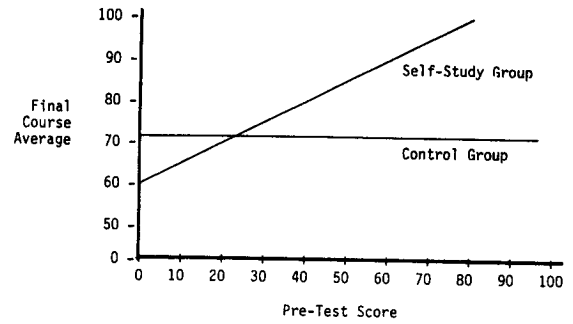


Figure 1: Final Course Average vs. Pre-test Score for Self-Study and Control Groups

CONCLUSIONS AND RECOMMENDATIONS

In terms of solving the basic problem of facilitating a large number of students in the Introduction to Data Processing course with very limited resources, the use of a Self-Study teaching methodology was quite successful. Of much greater interest, however, is the Self-Study methodology's impact on scholastic performance as compared to that of the traditional lecture methodology. As Figure 1 illustrates, those students who came to the course with little or no prior knowledge of computers exhibited a much lower relative gain in knowledge as a consequence of participating in the Self-Study program than did those students who began the course with a higher level of prior knowledge. Among those students participating in the traditional lecture program, there was no such discernible relationship between prior knowledge and scholastic performance.

As the goal of including an Introduction to Data Processing in the School of Business' common body of knowledge was to ensure that all Business majors acquired a minimum knowledge of computers, this study has shown that one group of students, those entering the course with little or no prior knowledge of computers, were not being adequately provided for by the Self-Study program. Their counterparts in the Control group exhibited a higher relative gain in knowledge. Consequently, it is recommended that, when a Self-Study teaching format is used, a traditional lecture program should also be made available, with assignment to one or the other being made based on prior knowledge of the subject. Such a "dual teaching methodology" program will conserve available resources while providing a structured environment for those students with little or no prior knowledge and, at the same time, allow those students with a higher level of prior knowledge to acquire additional knowledge at a greater rate than the structured environment can accommodate.

REFERENCES

1. Draper, N.R. and Smith, H., Applied Regression Analysis, John Wiley & Sons, Inc., New York, 1966.
2. Hick, C.R., Fundamental Concepts in Design of Experiments, 2nd Ed., Holt, Rinehart and Winston, New York, 1973.
3. Johnston, J., Econometric Methods, 2nd Ed., McGraw-Hill Book Company, New York, 1972.
4. Wildt, A.R. and Ahtola, O.T., Analysis of Covariance, Sage Publication, Beverly Hills, CA, 1978.

A TEAM PROJECT APPROACH TO TEACHING SYSTEMS ANALYSIS

Jean Buddington Martin
Assistant Professor of Computer Information Systems
Jacksonville University

Students are assigned to groups to work on problems associated with a case study highlighting concepts and terminology presented during an introductory class in systems analysis and design. Student and teacher responses to this method of instruction are summarized.

OVERVIEW OF THE COURSE

The traditional approach to teaching the introduction to systems analysis course at Charles County Community College, La Plata, Maryland, is to combine exposing the students to the role of a systems analyst in the analysis, design, development, and installation of computerized systems along with a practical experience in the systems development life cycle through the use of a case study. The rationale for this approach is to provide the students with basic terminology and concepts which are reinforced and expanded upon when preparing the case.

Major topics presented in this class include the systems study request, customer organization charts, reasons for system study requests, "resistance to change," systems flowcharts (considerable emphasis is placed on teaching students to use this tool), problem definition, tasks of each of the systems project steps (phases), project management (scheduling techniques, CPM, pert charts), design of output medium, design of input medium (includes some file organization concepts), and controls (system, group and individual record).

The author perceived a need to incorporate some new terms and tools as well as modifying the method of instruction in teaching the course. It seemed important to at least expose the students to alternative tools such as data flow diagrams and data base concepts.

Since later courses are dependent upon information received in this course, the author felt constrained not to deviate too far from the syllabus. Consequently these additional topics were introduced during the last class sessions of the semester. It is a difficult decision for teachers and those responsible for course/curriculum development to decide when it is time to make major changes in the content of courses particularly when the more traditional tools are still being used extensively in the subject area.

THE CASE STUDY

The text provides the commentary for the case study that is developed in the class. Additional information is provided as the chapters progress. Through the instructor's guide the teacher imparts further background data concerning the case. There is considerable opportunity to embellish and improvise within the case structure depending on the teacher's background and focus. An example case study in the text provides the students with many ideas to incorporate in their assignment.

RATIONALE FOR USING A GROUP APPROACH

The following factors contributed to the decision to take a group approach in utilizing the case study project: The author has previous teaching experience in the area of management development. One of the primary courses that she taught was principles of management, in which a case study approach was also used. The author made extensive use of groups to develop concepts in that class. Both the students and the instructor believed that the group work was very beneficial.

When preparing to teach the introductory systems analysis class, it was noted that the authors of the textbook suggested that one approach to the case study is to divide the students into small groups to discuss and implement the case. Because many of the students have very little previous coursework and very little work experience, many students are not individually prepared for the challenge of some of the assignments in the case study. As is typical in many community colleges, the composition of day classes consists primarily of students 18 or 19 years of age with minimal exposure to ideas and practices of industry.

Increasingly, industry is using a team approach for projects. Giving the students the formal opportunity of interacting with team members seemed an additional advantage to taking this approach in presenting the case study.

COMPOSITION OF THE GROUPS

There were 24 students in the introductory systems analysis class who were randomly divided into four teams of six members. After providing them with information and expectations concerning the assignments and allowing class time to meet with fellow group members, the instructor requested that each group select its own team leader. The group leader was to be responsible for the coordination of assignments and communication within the group.

STUDENT REACTION TO THE GROUP APPROACH

At the end of the semester the students were requested to complete a questionnaire which primarily centered around their perceptions of using a team approach on the project. Although they could answer the questions anonymously, they were requested to include the name of their leader. The following is a summary of their reactions based on the questionnaire:

- (1) Question: Would you recommend using a group approach again? Response: all except two students recommended that the approach be used.
- (2) Question: What recommendations would you suggest if a group approach were used again? Response: The following points were those most frequently reported: (1) Allow more class time for groups to meet (they cited difficulty in scheduling meetings at other times). (2) Have some way of penalizing those students that did not do their "fair share." (3) Require more prerequisite courses, including business courses. (4) Construct the project so that it would not be so time consuming. (5) Have smaller group sizes.
- (3) Question: What are the advantages of working on a team? Response: The following advantages were cited most frequently: (1) exposed to a greater variety of ideas, (2) lessened the individual work load, (3) obtained greater amount of expertise, (4) helped in learning to work with others to reach a common objective, (4) enjoyed the company of other students on the team, and (5) gave greater understanding

of how work might be accomplished in the "real world.

(4) Question: What are the disadvantages of working on a team? Response: The following disadvantages were cited most frequently: (1) had difficulty in finding the time for the entire group to meet together, (2) some students did not do their share of the work, (3) could not reach a consensus when there was a difference of opinion, (4) there were personality differences, and (5) there was a lack of leadership.

(5) Question: What are the differences in the manner the group worked on the second part of the project in comparison with the first part of the project (did they feel as though they worked better or worse)? Response: Of the 21 students responding, 12 perceived themselves working more productively on the second part of the project. The primary reason cited being that they knew each other's strengths and weaknesses better. Seven of the students perceived the group as being less organized when attempting the second part of the project. The primary reason cited was that assignments from other classes had become more demanding. One student reported no difference. It was difficult to ascertain the meaning of the response from one student. It should be noted that the responses to this question varied within each group.

INSTRUCTOR'S COMMENTS AND OBSERVATIONS

There were a number of benefits derived from taking a team approach to the project. Foremost of which is developing a more realistic view of the positive and negative aspects of working with others on a project. The students encountered many of the difficulties during this simulation that would confront them in a job setting. The necessity of learning to interact positively with other group members in order to accomplish goals was required in order to complete the project successfully. They experienced the problems of personality differences, apathy and unwillingness or inability of some team members to carry out their responsibilities to the group.

Two of the groups exhibited a minimal amount of confusion in meeting their objectives. Their morales appeared to remain high. While their first group presentations to the class were satisfactory, in both cases the second presentations were outstanding. Their organization, technique and quality of work improved greatly. Both of these groups were cohesive and supportive of individuals within the group. The other two groups did not demonstrate these characteristics. They did not give the impression of working nearly as well together. Neither group presentation was well planned or implemented.

Members of the two more successful groups (Group A and B) also worked on their individual assignments together. There was marked similarity in the output of these tasks. This was of particular interest since Group A members did not perform these tasks well. They all utilized the same ideas which were neither realistic nor correct. Group B members performed extremely well with these tasks. The conclusion is that if the group went in a direction that was suitable for the problem, all the members benefited. Likewise, if the direction was not suitable, all the members of the group fell into the same trap.

The members of the other two groups did not work together on the independent assignments. Their papers showed markedly less similarity. There were instances where it was apparent that two or three

students worked out the problems together, but the performance of these two groups spanned a much greater range in how successfully they completed the assignments. None performed as well as the members of Group B. When questioned as to who in Group B provided the ideas for the solutions, they responded that they had gotten together on several occasions (once until 2:00 a.m.) to work out the problems. They also reported that there was no one person who provided the answers, that they had all shared their ideas and contributed to working out solutions. Students were not penalized for working together on the independent assignments even though it was not the instructor's intention to have them do so. Once assigned to a group it seemed difficult for them to separate independent assignments from group ones.

It was further observed that the leader played an important role in how well the group performed. Both Groups A and B had very able, enthusiastic leaders who communicated frequently with their team members. Neither appeared to be autocratic in their methods, yet were able to maintain the quality of production of the group. Both did express their inability to delegate responsibility as well as they would have liked.

The leaders of the other two groups, although potentially as able as the leaders of Groups A and B, did not appear to be as motivated or enthusiastic as their more successful counterparts. Their groups were less cohesive and the morale of members seemed lower. Some of the individual members still performed well, but this appeared to be in spite of the group rather than because of the group. Just how much group leadership contributes to the group is not the focus of this paper, yet there did seem to be some relationship between leadership and the morale of group members.

Using a team approach can increase the student's awareness of the importance of group dynamics in accomplishing corporate objectives. This approach also seemed to improve the intuitive understanding of the concepts and terminology germane to the course. Group interaction was evident not only during their planned sessions, but in the classroom where the instructor often had to interrupt discussions about case studies in order to begin the day's instruction. Absenteeism was almost nonexistent.

The two major disadvantages of this approach which were cited by the students deserve comments. First, the problem of arranging meetings can only be reduced by letting students try to organize their own groups. It should be noted that this would not be representative of a real life situation. Second, the problem of 'penalizing those who do not carry their fair share' can be addressed by permitting the students to assign a range of points to team members based on perceived contributions.

Finally, it seems appropriate that serious consideration should be given to establishing 'Cobol' and either 'an introduction to business organization' or an introductory accounting course as additional prerequisites.

THE LIVE STUDENT SYSTEMS DEVELOPMENT PROJECT

Iza Goroff
University of Wisconsin-Whitewater

The live student project is compared to the model student project. Project control techniques are described which can overcome the difficulties inherent in the live student project.

The purposes of this paper are:

- to present the advantages and disadvantages of the live or "real" student systems development project over the model project
- to show how to overcome the disadvantages.

Both the DPMA Model Curriculum for Undergraduate Computer Information Systems Education (1) and the ACM C.I.S. Curriculum (2) include core courses requiring student systems development projects. Most systems analysis and design course sequence (3,4,5) require that the student, as a member of a team, develop either a model project or a real (live) project.

The Live Project vs. The Model Project

The model project has the following advantages:

- client interviews are controlled (i.e., actually given by the teacher to make specific pedagogical points)
- the system and its goals are well defined
- the system can be carefully chosen to demonstrate the principles taught
- all students have the same uniform experience
- it is easier for the teacher to keep track of one system
- it is easier to compare student achievement and assign a grade
- there is no shortage of systems to be designed.

All of these advantages come down to one: the control that the teacher has over the course and the system to be designed.

The live student project has all of the corresponding disadvantages:

- clients behave unpredictably
- system goals are unclear
- system requirements may not be what was expected
- students may differ in what they learn, depending on what roles they play and on which project they are placed
- the teacher will have from 3 to 7 students/project resulting in a large number of projects to follow
- there may be a shortage of available projects.

The teacher must work much harder for what is a non-uniform educational experience, so why do it?

The goal of systems analysis is the discernment of order in what appears to be confusion. A model system offers too selective a set of "facts" to give much of the feeling of the analysis process. Systems design includes the establishment of order into a real environment. That part of the design process is missing from the model system approach. In short the "disadvantages" listed above are mostly advantages insofar

as the education of the student is concerned. Quirky clients, unclear systems goals, unpleasant surprises, and other manifestations of "reality" are what students must learn to deal with, especially in the relatively low risk environment of the university setting.

However, we believe that it is wrong to rely completely on the experience of a single project to ground students in the principles of information systems. These projects are performed in the context of a two course (one year) systems analysis and design sequence (3) where there are non-project assignments to guarantee coverage of the basic principles of analysis and design.

MANAGING THE LIVE PROJECT

Acquiring Projects

Almost all of the projects have come from administrative areas of the University. Systems have been developed for the placement office, library, student health center, tutorial center, alumni center, ticket office, athletics department, among many others.

Most campuses have a newsletter for staff communications which can be used to advertise the availability of "free" systems development. Once a client has a successful system, that client may want others developed, as well. The university's director of information systems development usually has a list of projects whose priority is so low that it is inconceivable that they would ever be attempted, which the director may happily share with you.

We project a three year life cycle for the student developed systems. A system designed three years ago is ready for re-design, giving a continuous supply of systems for our student teams.

Team Selection

The projects are listed on the Project Selection Form. Each student ranks the projects from 1 to 8 or more, based upon interest in the project. Subject to the constraints of optimum team size and keeping a team entirely within a class, we try to follow that prioritization. Most students receive their first choices.

The team chooses its own leader. Occasionally, the leader is replaced during a semester. We encourage the election of a new leader at the start of the second semester.

Project Control

The control of a number of projects, each employing a team of from three to seven students, is a formidable challenge to the teacher. Project control by the teacher is much like that exercised by a manager of information systems who controls several projects and project leaders. The remainder of this paper is devoted to discussing the structure we have established to maintain control over the progress of the student project.

Our first attempts in controlling student projects had an unacceptably high failure rate. Only when we adopted procedures and principles used in the more successful information systems development areas of business that projects have been consistently completed. These principles are:

- . no project task is considered complete until it is right
- . documentation is structured by standard forms, the same forms applying to all projects
- . documentation is part of every task; there are no separate documentation tasks
- . full credit is given only when the task is completed on time, according to the pre-set schedule given at the start of the course.

The forms we use were developed as part of a student project by William Star, Brian Ney and Richard Peltier, who together also developed an on-line information system for the control of student projects. These forms are based on a documentation system developed by Touche-Ross & Co. (6), but they are adapted to the systems design methodology used (3). The project task schedule for the first semester, including the analysis and logical design phases of the project, follows in Table 1. The tasks in this semester are sequentially related.

TABLE 1 PROJECT TASKS - FIRST SEMESTER

<u>Task #</u>	<u>Date</u>	<u>Task Name</u>	<u>Documentation for Completion</u>
1	9/16	team selection	Project Team Information Sheet (#100)
2	9/23	user contact 1	Interview Summary (#120)
3	10/11	project definition	Project Definition (#130)
4	10/22	entity/attribute enumeration	File Contents (#190) partially filled out
5	10/22	user contact 2	Interview Summary (#120)
6	10/29	analysis of existing system	Narrative (on blank form #150)
7	11/1	preliminary systems proposal	Narrative (task 5) File Contents (task 4) Procedure charts & explanations (#150)
8	11/8	user contact 3	Interview Summary (#120)
9	11/22	sample screens, -output	Samples (#150)
10	11/22	final systems proposal	Narrative (revised) File Contents (revised) Procedures (revised) charts explanations sample conversations & output (task #7)

The project task schedule for the second semester, the physical design and implementation phases of the project follow in Table 2.

The precedence relationship of the tasks is more complex than in the first semester. This is shown in the network diagram in Figure 1.

While some projects may require tasks beyond those listed in Tables 1 and 2, the completion of the listed tasks will usually force those other tasks to be finished as well; i.e., it is unlikely that the listed tasks could be completed without the completion of the rest of the requisite tasks.

TABLE 2 PROJECT TASKS - SECOND SEMESTER

<u>Task #</u>	<u>Date</u>	<u>Task Name</u>	<u>Documentation for Completion</u>
1	1/24	revise logical design	replacement pages for final system proposal
2	1/24	choose computer, operating system, language	computer/operating system/language page (blank form)
3	1/26	physical file design	preliminary data dictionary with variable names
4	1/26	request user to provide data for table files (i.e., semi-permanent data)	interview summary form signed by user - 10 points only
5	1/26	distribute logical procedures among team members	blank form
6	2/3	review/learn operating system, language	simple direct access programs to read, write and update test files
7	2/7	develop test data and expected outcome for each logical procedure	test data (input) listing, initial and final (expected) file contents
8	2/21	design structure charts	structure charts
9	2/21	report progress to user	interview summary signed by user - 10 points only
10	2/23	walkthrough of structure charts	structured walkthrough form
11	2/28	revised structure charts with utilities identified	structure charts
12	3/1	module specifications	module specification sheets (partially filled in)
13	3/6	pseudocode for first procedure's modules	module specification sheets (partially filled in)

TABLE 2 (continued)

<u>Task #</u>	<u>Date</u>	<u>Task Name</u>	<u>Documentation for Completion</u>
14	3/8	develop test data and expected outcome for each module for first procedure	test data sheet containing values of input parameters, stub arguments, expected output parameters
15	3/8	pseudocode walkthrough for first procedure	structured walkthrough form
16	3/8	revised data dictionary	revised data dictionary
17	3/15	code modules for first procedure	program listings from computer printout
18	3/22	code test drivers & stubs for first procedure	program listings from computer printout
19	4/5	test modules for first procedure	(initial file listings), test program run with printout (final file listings)
20	3/20	report progress to user	interview summary form signed by user - 10 points only
21	4/10	test first logical procedure	(initial file listings), logical procedure run with output, (final file listings)
22	4/12	pseudocode remaining procedures	see task 13
23	4/12	test data & expected outcome for each module of the remaining procedures	see task 14
24	4/12	pseudocode walkthrough for remaining procedures	see task 15
25	4/17	code modules for remaining procedures	see task 17
26	4/19	code testdrivers, stubs for remaining procedures	see task 18
27	4/24	test modules for remaining procedures	see task 19
28	4/26	test remaining logical procedures	see task 21
29	5/1	user documentation	indexed for each logical procedure: revised logical design, sample runs, recovery procedures
30	5/1	program documentation	organization (with index) of all documentation developed on project
31	5/3	user training	interview form(s) with user, signature(s)
32	5/8	conversion - user training	user evaluation form
33	5/10	fine tuning	user signcfff

Grading the Project

We use four different measures in assigning a grade to a student:

- . individually graded oral presentations, all in the first semester
- . a set number of points/task completed on time, 10% penalty/day late down to a salvage value of 20%
- . teammate evaluation (7)
- . client evaluation

The points given for tasks completed on time are a minor part of the first semester's grade but about 50% of the second semester's. These points are given only if all of a task, that of the whole team, is complete. Laggards are dealt with in the teammate evaluations.

Results of the Project Control Methodology

The results using this methodology for the academic year 1983-84 are shown in Table 3.

TABLE 3 RESULTS OF PROJECT CONTROL METHODOLOGY

<u>System Name</u>	<u># of Students</u>	<u># of Logical Procedures</u>	<u># of Interrelated Files</u>	<u>Lines of Code</u>	<u>Completion Status</u>
Intern Opportunity	5	8	8	2,500	late
Book Ordering for the Blind	4	5	3	2,000	on time
Athletic Scheduling	7	9	12	3,500	on time
Government Documents	5	6	3	3,000	late
Farm Marketing	6	8+	6	4,000	on time
Crime reporting	5	12	10	3,500	partial
Folksong-Kodaly	5	8	6	4,000	on time
Child Care Center	7	12	12	5,000	on time
Bid Tracking	7	10	11	4,000	late

All projects listed as "late" were still completed within the semester. Incomplete systems have four possible fates:

- . staying incomplete
- . completion by team members after the semester's end, possibly in return for a grade change
- . completion by "Independent Studies" students
- . redone by a new team, a year or more late.

While completion is strongly encouraged, clients are told at the start of the process that there is possibility of failure.

The results shown above represent a marked improvement in completion over previous offerings of the course where the on time completions were less than 10% of the number attempted. The number of late or partial completions is still higher than desirable. These almost always result from the failure of one to three students to keep pace with their teammates. Most frequently one of these failures is caused by the personal problems of the students, what might be ground for firing in a real job situation. Even the failures contribute strongly to the student's educational experience: understanding how the success of a project depends on one's teammates is considerably increased.

Conclusion

The live student project is an educational tool of real value. Close control of the project is necessary for the likely achievement of both its academic and system's goals.

BIBLIOGRAPHY

1. Adams, D. R. and T. H. Athey, DPMA Model Curriculum for Information Systems Education. DPMA, 505 Busse Highway, Park Ridge, IL 60068.
2. Nunamaker, J. F., Jr., J. Couger and G. B. Davis (editors), Information Systems Curriculum Recommendations for the 80's: Undergraduate and Graduate Programs, Communications of the ACM, Vol. 25, No. 11, pp. 781-805.
3. Goroff, I., A Systems Analysis & Design Course Sequence, ACM SIGCSE Bulletin, Vol. 14, No. 1, pp. 123-127.
4. Golden, D. G., Development of a Systems Analysis and Design Course, ACM SIGCSE Bulletin, Vol. 14, No. 1, pp. 110-113.
5. Barrett, R. A., A Five Course Sequence for Information Systems, ACM SIGCSE Bulletin, Vol. 14, No. 1, pp. 114-122.
6. Biggs, C. L., E. G. Birks, W. Atkins, Managing the Systems Development Process, Prentice-Hall, (1980).
7. Ref. 3, page 126

Iza Goroff
 Management Computer Systems Major
 UW-Whitewater, Hyer 309
 Whitewater, WI 53190

(414) 472-1468

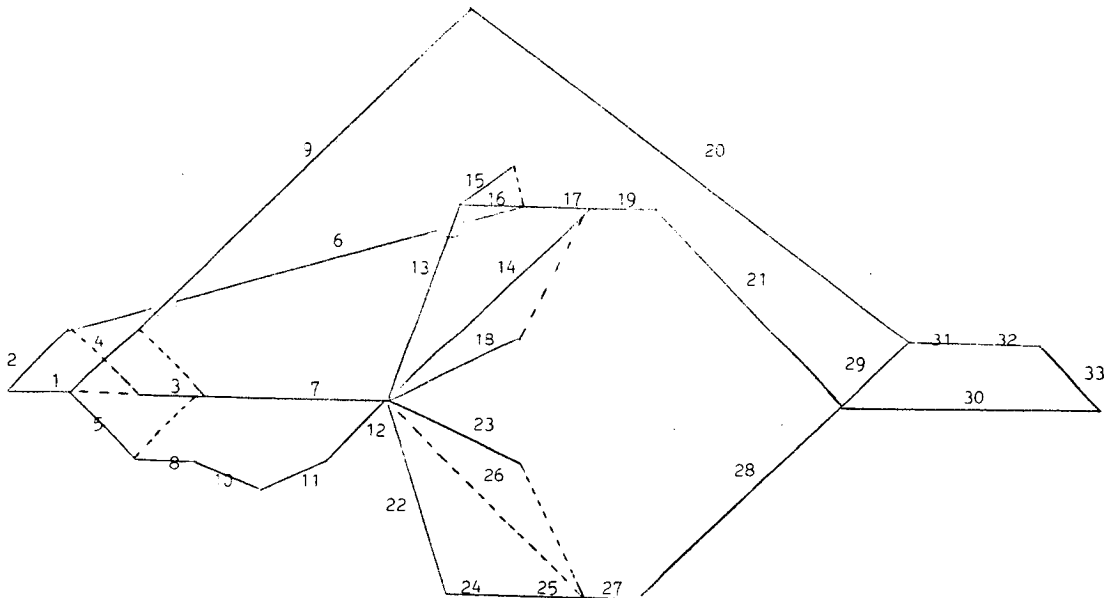


Figure 1.

THE ROLE OF MICROCOMPUTERS IN THE CIS CURRICULUM

Wayne D. Smith
Department of Computer Studies
Murray State University
Murray, KY 42071

This paper presents a case for integrating microcomputer instruction throughout the courses in the CIS curriculum. This approach is recommended in lieu of the more traditional single course in microcomputers. Arguments favoring a spiral approach to teaching microcomputer topics is given. Some discussion is presented indicating where certain microcomputer topics would be appropriate in the current CIS curriculum course guidelines.

HISTORICAL PERSPECTIVE

Much of the current Computer Information Systems curriculum was developed before the widespread availability of microcomputers. As a result, microcomputers do not receive a great deal of attention in the curriculum guidelines. Most CIS courses are related to the use of mainframe computers. This situation is not inappropriate, since most data processing is currently performed on this type of computer.

On the other hand, the rapid growth of the use of microcomputers in business promises to bring about significant changes in the way the business community views computers. The Computer Information Systems curriculum of the 1980's cannot afford to ignore a tool with the microcomputer's potential. Fully defining the role of the microcomputer in the CIS curriculum will be an important undertaking that will require attention for the remainder of this decade.

When new techniques are developed in the business world, the academic community has traditionally responded by adding new courses to the curriculum. This is the current situation in CIS where a profusion of "microcomputer" courses are being offered to CIS students. Such courses have serious shortcomings, and are probably not the best approach to teaching microcomputer concepts to the CIS major.

WHERE MICROCOMPUTERS BELONG

There are several disadvantages associated with providing any single "microcomputer" course in the CIS curriculum. Teaching a course on microcomputers seems to imply that the microcomputer is an entity that can be studied in isolation. This is definitely not the case, since microcomputer usage crosses the boundaries of many of the more traditional CIS subjects. Further, no single course can possibly address all the important implications of the use of microcomputers in business. In short, the microcomputer is far too important to be relegated to a single course.

Instead of a single course on microcomputers, the CIS major needs repeated exposure to microcomputer concepts presented in the environment where microcomputer utilization will be most advantageous. Pedagogically speaking, the CIS major needs to be taught about microcomputers through the use of a spiral approach. The student should be exposed to the microcomputer at increasing levels of complexity

throughout the curriculum. Each exposure should build on previous knowledge while stressing new application areas.

The spiral approach offers several distinct advantages. The student will be exposed to the microcomputer at a purely practical level early in his educational career. This will familiarize him with the capabilities of the microcomputer and, at the same time, provide him with tools such as word processors and spread sheets that he can use in subsequent courses. In later classes, he will receive more detailed instruction about the microcomputer and its capabilities. This will enable him to evaluate the applicability of the microcomputer to the solution of a variety of business problems. The CIS graduate of the 1980's will be able to utilize the microcomputer both as an administrative tool in his own work and as an instrument for providing user services within the business organization.

MICROCOMPUTERS IN CIS COURSES

The following discussion indicates some methods that might be used to integrate microcomputer material into the CIS curriculum. These ideas are not intended as a detailed plan of attack, but are proposed as a point of departure for further planning. The important point is that microcomputers should be introduced to the CIS student very early in the program, and this initial exposure should be reinforced throughout his educational experience. In most cases, the individual faculty member responsible for each course will be the determining factor as to the amount and type of microcomputer exposure presented in a specific course.

CIS-1, Introduction to Computer Based Systems. The CIS student should be introduced to the microcomputer during his first course. This introduction should cover the basic principles of the microcomputer, and the lectures should be supplemented with laboratory exposure. This exposure should provide a basic introduction to operating systems, word processors and spread sheet applications software. If time permits, there can also be some introduction to data bases, file handlers, report generators or microcomputer programming languages.

There are two reasons for the early introduction of this material. By using this approach, the student will be given a brief "user level" introduction to the microcomputer and applications software that will provide him with practical tools for use in other CIS

and business core courses. At the same time, the student will be introduced to the concept of the microcomputer as only one of many types of computers that can be used to satisfy business data processing needs.

A number of textbooks are beginning to appear that provide materials to support a course of this type. Publishers are including some applications software as part of an overall materials package for this course. This may include either a scaled down version of a commercial package such as Framework, or a "generic" package written specifically for that publisher. In either case, these materials are usually adequate for use in classroom exercises.

CIS-2&3, Applications Program Development I and II.

In the two Applications Programming Development courses, there is probably little advantage to using microcomputers. The primary purpose of these courses is to develop the student's basic programming skills. The type of computer used for this purpose is immaterial, since it serves only as the means to an end. In most cases, a mainframe computer with large scale data handling capabilities will be most appropriate. In cases where adequate microcomputer facilities are available, these can be used to demonstrate alternatives to the mainframe computer.

CIS-4, Systems Analysis Methods.

The systems analysis course offers an excellent opportunity to present many microcomputer concepts to the CIS student. Microcomputer applications should be examined along with the more traditional manual, mini-computer and mainframe applications. The student should be exposed to some examples where the microcomputer presents a cost effective alternative to other types of information processing systems.

This course should also be used to further the student's understanding of the capabilities and limitations of microcomputers. Commercial business applications software should be examined, with special emphasis on dedicated systems. The student should also be exposed to techniques and procedures for testing and evaluating microcomputer hardware and software.

CIS-5, Structured Systems Analysis and Design.

The Structured Systems Analysis course offers another opportunity to integrate microcomputer concepts into the CIS curriculum. It is relatively easy to develop examples and case studies where the microcomputer offers an effective solution to a particular business problem. It might be possible to devote the early portion of the class to a small design problem that would utilize microcomputers with commercial applications software. Later portions of the class could concentrate on designs for larger systems on mini-computers and mainframes.

CIS-6, Database Program Development.

The use of microcomputers offers an excellent method for introducing the concepts of database systems. By utilizing a microcomputer based system, the principles and concepts can be introduced without the overhead associated with large scale systems. In addition, the microcomputer system can be used for early assignments while the more complex mainframe system is being introduced in lecture. The microcomputer database system would supplement rather than replace the mainframe system. The well prepared CIS student should be comfortable with both types of systems.

CIS-7, Applied Software Development Project.

In the Applied Software Development Project, students have traditionally been concerned with developing a large scale, mainframe based systems. However, it would be quite feasible to integrate microcomputers into this course. If microcomputers are not used for the main design, they could be incorporated into the larger system as point-of-transaction data collectors, data transmission nodes, intelligent terminals, etc.

Other CIS Courses. The preceding discussion indicates how the microcomputer can be integrated into the core of the CIS curriculum. There are many other courses where the microcomputer can be used to advantage.

In CIS-8, Hardware and Software Concepts, microcomputers could be used as the vehicle for examining both computer hardware and software. The relative simplicity of microcomputers make these machines ideal for this purpose. Both microcomputer assembly language and hardware interfacing topics can be covered.

In CIS-9, Office Automation, the microcomputer provides an excellent vehicle for examining word processors, spread sheets, filing systems, and integrated software. Microcomputer networks can also be discussed in this course.

In CIS-10, Decision Support Systems, microcomputers can be used as management tools to support decision making. Filers and simple data base systems are a natural in this area.

CIS-12, Distributed Data Processing, would benefit from an introduction of microcomputer material. If microcomputer networks have not been introduced previously, this course presents a good opportunity.

In CIS-15, Resource Management, the management of microcomputer resources should be discussed. The problems of data base duplication, purchasing control and data security pose particular problems in relation to microcomputers.

In a well rounded CIS program, the microcomputers should also be integrated into other courses in the general business core. This would include courses in Accounting, Management, etc. By introducing the microcomputer in the early CIS courses, the use of microcomputers in other courses will be encouraged.

MOVING TOWARDS THIS GOAL

While integrating the microcomputer into the entire CIS curriculum will require considerable effort and resources, it offers the most promise for preparing the CIS student to effectively utilize this tool in his profession. Unfortunately, few institutions have the resources to implement a program of this type in a single step. In most cases, the program will have to be implemented in stages.

One attractive method for integrating microcomputers into the CIS program would be to use a top down approach. That is, integrate microcomputers into the upper division courses first. This approach will provide current students with some exposure to microcomputers and still minimize the extra workload imposed on faculty. It will, however, result in a restructuring of the course content in order to accommodate the introductory microcomputer materials.

Even if the top down approach is used, it would still be advisable to integrate the microcomputer into the first course at the earliest opportunity. As additional upper division courses are modified, the students enrolling in these courses will arrive with some prior exposure to microcomputer concepts. This will reduce the amount of course content restructuring that must be undertaken to integrate the microcomputer into these courses. As mentioned earlier, this will also provide the students with tools that will be useful in other courses.

Regardless of the means used, the eventual goal should be the complete integration of microcomputers throughout the CIS curriculum. Only in this fashion can the CIS graduate have the expertise to view the microcomputer in its proper perspective as an effective and economical tool for meeting the data processing needs of the business community.

DEVELOPING MICRO-BASED COURSES

An Information Systems Model

Thomas E. Burke
Bryant College

ABSTRACT

In September 1983 Bryant College launched a new MBA course "Micro-based Business Systems." Its main objective is to introduce use of the personal computer in managing small to medium-sized businesses. Having completed four "rounds" of continuous development, a delivery model has evolved that integrates case analyses, classroom lectures, computer lab exercises and assignments, plus a capstone term project. Since the initial offering, all sections have been fully subscribed well in advance...a modest success story!

Touching on procedures followed in a team development effort, the paper focuses on the importance of an information system for planning and managing courses involving active use of microcomputer hardware and software. Features of the information system that evolved and worked well at Bryant are described. The role of student projects is also emphasized.

ISSUES and CHALLENGES

By sharing experiences and hard-earned insights, it is hoped the paper will be of value to other professional educators who face similar challenges to what we encountered in our innovative efforts. These included:

1. Team Development

How do we **integrate** the creative talents of a faculty team in pulling together a diverse set of tutorials and exercises? To what degree should one establish detailed design criteria?

2. Organization and Sequence

To what **level of detail** do we cover specific topics, such as system software, accounting, strategic planning, data base management, etc.? Each member of the development team had his/her own ideas! In what **sequence** should we introduce the various applications software packages and techniques?

3. Theme and Substance

What provides the **substance** for integrating the course? We were concerned with the amount of time it might take to familiarize students with the "how-to" of applications software. In attempts to accommodate a range of backgrounds, our fears were often realized.

4. Communication Base

From the student's perspective, getting introduced to the fast-changing world of microcomputers can be an overpowering, and often frustrating experience. The student is suddenly called upon to deal with a variety of manuals, software packages, templates, formats, assignments, due dates, unfamiliar terminologies, etc.

In that context the professional CIS educator has to do a super job in describing goals and steps needed to carry out these goals. Essentially, we are talking about the importance of **management and effective communication** in a highly technical environment.

In addition, there were many other areas which demanded careful attention. Not discussed in the present paper, these include: selection and funding of hardware and software, maintaining security, grading, scheduling lab time, and "software obsolescence."

Faculty teaching the course still do not have answers to all these questions. However, our pilot efforts have been most successful in gradually building an **information system** that does address the four crucial areas of concern.

As displayed in Exhibit A and detailed on the following pages, the system comprises six main components, each "linked" to one or more of the previously mentioned critical areas.

INFORMATION SYSTEM COMPONENTS _____ Exhibit A

Issues and Challenges.....			
	1	2	3	4
	Team Develop.	Organization and Sequence	Theme and Substance	Commun- ication
■ Course Design	*		*	*
■ Terms	*	*		*
■ Block Plan	*	*		*
■ Unit Spec	*	*		*
■ Notes/ Cases	*		*	*
■ Final Project			*	*

* Course Design and Development

In June 1983 the author took on the task of integrating written materials and software being developed by four other faculty at Bryant College. With the course set to start September 1983, there was obvious need for some type of uniform format to guide each team member in his efforts. The procedural model, Exhibit B, was followed. In conjunction with the Block Plan, it served as a valuable reference for communicating within the "Micro-Team."

* Terminology

Early in the project we established a common set of terms and conventions to facilitate the editing process. Exhibit C is part of a one page document that proved valuable for both the faculty and students as well.

* Block Plan

As shown in Exhibit D, the Fall '84 offering divides the course into five major Blocks, each dealing with a different set of topics and associated software. Use of a business graphics package makes it easy to update the schedule for subsequent semesters.

*** Unit Specification**

If the author were asked to pick the most useful aid for managing this type of course, it would be the Unit Specification.

Each individual module, corresponding to one week of activity, was introduced with a one or two page hand-out. As an illustration, Exhibit E was issued prior to the start of our first session on spreadsheets. The format is quite straightforward and can be easily tailored to meet the needs of any course.

It was found that these are the main items needing definition:

- *Title
- *Purpose Of Unit
- *Required Skills
- *Lab Exercise(s)
- *Software/Templates
- *Tutorials/References
- *Assignments
- *Due Dates

***Tutorials and Cases**

As it worked out we were able to select one case "Chalet Susse International" (Harvard 9-673-090) that provided a central theme for the Strategic Planning Block. Financial data in the case was used to prepare pro-forma income statements, sensitivity analyses, graphical plots, and data base manipulations.

Since we were unable to locate an acceptable text for the course, it was necessary to prepare a series of Tutorial Notes. Each varies in length from three to five pages and focuses on "generic" topics associated with the course and case selected. Representative examples would include: Modeling, Text Management, and Marketing. Where possible, we tried to write these independent of the software packages being used in the course. The majority of documentation was prepared using a word processor. That decision proved invaluable for future updates of course materials.

*** Final Project**

Faculty teaching the course have given top priority to the final project assignment. Initially we called for a final report to a hypothetical "loan officer," integrating various financial sheets and graphics generated during the semester. Most students did not feel challenged with the assignment.

Fortunately, a much better model met with success these past two semesters. This time we required the design, production and testing of an original interactive software package, where possible, to serve a useful function in the student's place of employment. Topics chosen were interesting and, despite initial reservations on the part of students, the results were outstanding!

As might be expected, the most common theme involved design and analysis of financial statements. Others dealt with topics such as sales commission systems, evaluation of advertising media, and cost versus profit of various sizes of paramedical offices.

Consistent with the information system theme, students are required to produce two documents in conjunction with the project.

1. Project Proposal...one or two pages submitted around the fifth week of the semester. Consistent with standard format the proposal defines project objectives, a software design approach, spreadsheet features to be employed, and the plan for user testing and evaluation.
2. Final Report...an expansion of the proposal outline, describes technical functions of the customized package, user guide, sample output, and validated checkout. Heavy emphasis is on professional quality and impact.

CONCLUSION and COMMENTS

* The individual components of the information system are quite straightforward. What may be most significant is their integration into a structure that facilitates continuous improvement of the course. Unfortunately, we did not have such a framework at the beginning.

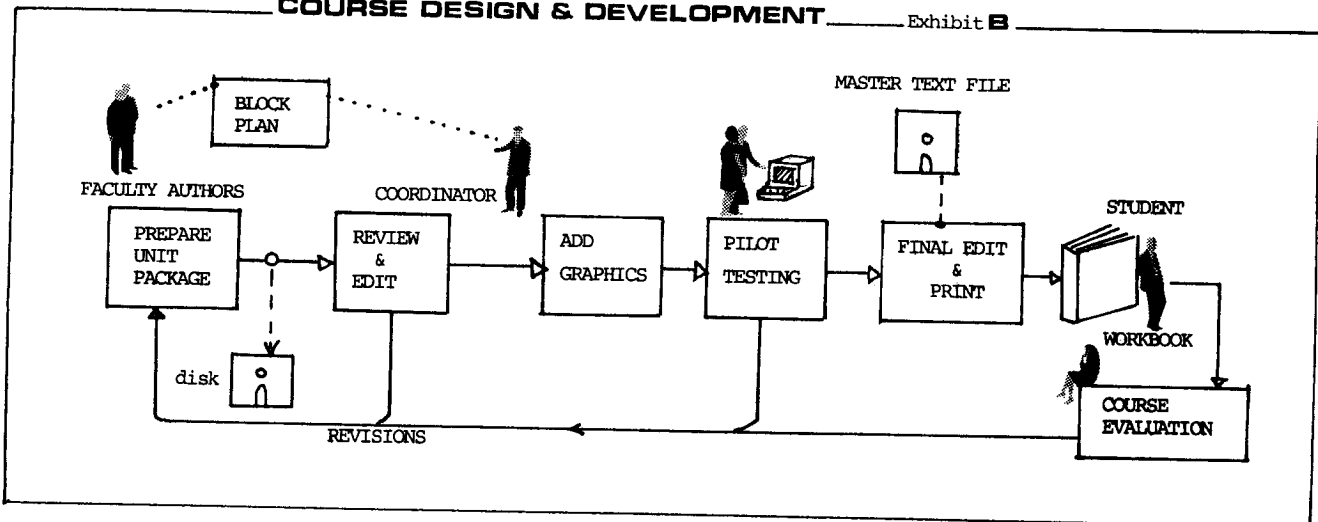
* Case analysis plus the tutorials and final project proved effective in focusing the key concepts and techniques covered during the semester. Together they provided the adhesive we were concerned about. Also, the project gave an open-ended opportunity for experienced students to use some of the advanced software features such as keyboard macros.

* Rather than depend entirely on vendor manuals and demonstration disks, emphasis was placed on the generation of written materials that lead the student step-by-step in getting familiar with the software. One major drawback is the amount of time necessary to put these tutorials together. Another problem arises when one decides to change a vendor package and you are forced to make major revisions in the handouts. Again, here is where a good word processing system is of value.

ACKNOWLEDGEMENTS

The author is sincerely grateful for the financial support extended by Peat, Marwick, Mitchell and Co. during the early phases of the project. Also special thanks are due my colleagues whose energies and materials provided a foundation for the course.... William Bygrave, George deTarnowsky, Francis Gathof, Alan Olinsky, and Wallace Wood.

COURSE DESIGN & DEVELOPMENT Exhibit B




TERMINOLOGY Exhibit C

This note describes how we have structured the course and defines some of the terms used in the handout materials.

The **COURSE** has been organized into major **BLOCKS**, each of these is made up of **UNITS**. An individual Unit includes these elements:

1. Lecture or Briefing
2. Demonstration(s)
3. Lab Exercise...not graded
4. Assigned Projects..completed during the week
5. Hand-out materials..tutorials, guides, etc.

You will be provided a **UNIT SPEC** which shows our plans for the coming period...usually covering one week of activity.

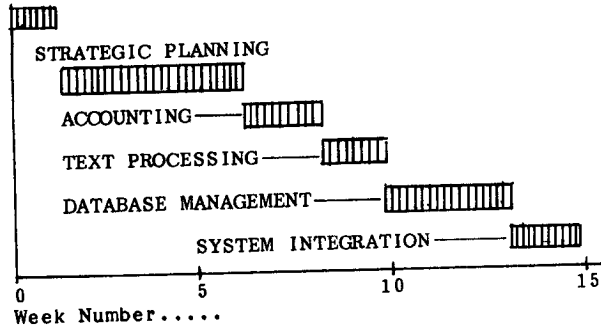
In some cases we will refer to the floppy diskettes as **DISKS**. In our diagrams a Disk is shown with the symbol 

Etc.

BLOCK PLAN Exhibit D

MICRO-BASED BUSINESS SYSTEMS...Fall 1984

GETTING STARTED



UNIT SPECIFICATION Exhibit E

INTRODUCTION TO ELECTRONIC SPREADSHEETS

PURPOSE

To become more familiar with the function(s), design, and modification of **Electronic Spreadsheets**.

SKILLS

Through the previous Exercises and LOTUS Tutorials you should now have a good feel as to the layout of a spreadsheet and some of the more basic commands needed to enter numerical information and functional relationships.

The present Unit will get you started with two "warmup" projects requiring most of the techniques needed for analysis of the Chalet Suisse case.

DEMONSTRATION

Vineyard Savings is a simple illustration of how the spreadsheet can be used for displaying financial data. Column H indicates the net savings of purchasing building materials on the "mainland" versus from lumberyards on the Vineyard.

LAB EXERCISE

Either by yourself or with a partner set up the spreadsheet called **Fruit Salad**. You may do this without changing column widths...the default setting is OK. Please note the use of **Formulas** for Column and Grand Totals. Also, try to figure out how we arranged the three names right-justified. Be creative in picking the names and in designing your spreadsheet. When you are pleased with your creation print one copy similar to the attached Exhibit. Hand in before closing shop for the evening.

ASSIGNMENT

1. Clip out a Local Stock Quotation from a local newspaper or the Wall Street Journal. See sample on next page.
2. Set up a table for at least six or seven stocks. You will need column headings and values for the Stock, High, Low, Sales, Close, etc. Please feel free to add other columns as appropriate.
3. Each column will show either a total, average, or whatever. You will use the Lotus functional and/or Formula techniques here. Perhaps you might also devise other useful calculations from the data...again, be creative.
4. Please include a brief introduction at the top of the spread sheet...Title, your Name, a statement indicating what your spread sheet is showing, etc. Three or four lines should suffice.
5. Make sure you save your work under the file name **STOCKS**. Print and hand in one copy at the start of class on.....

Author: Thomas Burke

PROFILE OF INTRODUCTORY DATABASE COURSES OFFERED AT AACSB-ACCREDITED INSTITUTIONS

Joan K. Pierson, Associate Professor, Kansas State University
 Jack Shorter, Assistant Professor, Emporia State University
 Jeretta A. Horn, Assistant Professor, Oklahoma State University
 G. Daryl Nord, Associate Professor, Oklahoma State University

ABSTRACT

A profile of introductory database courses taught in institutions accredited by the American Assembly of Collegiate Schools of Business has been compiled from the results of a survey conducted during the 1984 spring semester. The majority of the responding schools do offer an introductory course in database processing. In all but a very few institutions, enrollment in the database course depends on the completion of prerequisite courses. The majority of institutions which offer the course include it in the requirements for at least one degree program. Most schools assign project work on mainframes, mini, and/or microcomputers. A wide variety of computers and database management systems are used, the most popular being the IBM PC and dBASE II. Most introductory database courses include the topics recommended by the Data Processing Management Association in its model curriculum. Database courses are most often taught at the upper division or graduate levels by faculty who have both work experience using database management systems and a doctoral degree.

INTRODUCTION

This paper reports the findings of a 1984 study of introductory database courses in schools accredited by the American Assembly of Collegiate Schools of Business. The study was undertaken to gather information on:

- (1) the number of AACSB-accredited institutions offering an introductory database course,
- (2) course administration and enrollments,
- (3) hardware and software used,
- (4) course content, and
- (5) professional and educational background of the teaching faculty.

Twenty years ago, in 1964, the first significant database management system appeared. Database technology has progressed during the last two decades, and today database management systems are the backbone of many management information systems. Two professional organizations, the Data Processing Management Association (DPMA) and the Association for Computing Machinery (ACM), have recognized the importance of courses in database processing in degree programs and have included the course in their recommended curricula for business school programs in computer information systems. The two model curricula were developed in response to a need for guidance from professionals.

A majority of AACSB-accredited institutions are now offering either a degree program or a major emphasis in computer information systems. A survey conducted during the 1981-82 academic year revealed that of the accredited schools responding to the questionnaire, 59.7 percent had programs in place--and that 21.6 percent planned to offer a degree program or major emphasis in a degree program within the next three years. The same research listed courses on database concepts as the third most frequently offered course in schools with existing programs, behind courses in (1) systems design and analysis and (2) COBOL programming.

Another survey of AACSB-accredited institutions conducted during the spring of 1983 revealed that 61.1 percent of the responding schools offered a degree program or major emphasis in computer information systems during the 1982-83 academic year, and 22.9 percent planned to institute such a program within three years.

It appears that the number of AACSB-accredited institutions offering computer information systems programs is increasing and that introductory database courses are an integral part of the programs.

METHODOLOGY

A questionnaire was sent to the deans of the 236 schools accredited by the American Assembly of Collegiate Schools of Business in January of 1984. The deans were requested to forward the questionnaire to an appropriate faculty member for completion. A follow-up letter was sent in April, 1984, to those institutions which had not responded to the initial request.

The survey instrument included a question regarding the curriculum used for a computer information systems program--if such a program was offered. An inquiry was also included on the existence of database courses in the business school and/or

outside the business school. Those institutions having a business school database course were queried on administrative and enrollment data, hardware and software, content, and professional and educational background of the teaching faculty.

RESULTS

The findings of the survey are given in this section. A total of 159 questionnaires were returned--a response rate of 67.4 percent. The number of responses and the percentage of total responses for each question are given in the tables. The findings regarding introductory database courses reflect information on courses taught in schools of business. Data regarding database courses taught in departments or schools other than business are not included in this paper.

INSTITUTIONAL DATA

Table I lists the responses to a question regarding the type of curriculum followed for computer information systems programs in the institutions. There were 154 valid responses to this inquiry. Sixteen percent indicated that their schools do not have a computer information systems program. The curriculum most frequently noted (33.8 percent) was one which had been developed "in-house." The curriculum developed by the DPMA was the second most frequently noted (18.8 percent), followed by the curriculum approved by the ACM (13.0 percent). It is interesting to note that 11.7 percent of the responding institutions with computer information systems programs follow a curriculum which includes features from both the DPMA and the ACM models.

Sixty-six percent of the institutions which responded to the questionnaire offer an introductory database course in their business schools. Database courses are offered in other departments or schools in 60.6 percent of the responding institutions. (Some of the institutions offer the course both in the business school and in another department or school.)

Table I

INSTITUTIONAL DATA

	Number of Responses	% of Response
Computer Information Systems Curriculum Followed:		
No CIS program	25	16.2%
Developed by institution	52	33.8
DPMA Model	29	18.8
ACM Model	20	13.0
DPMA & ACM combination	18	11.7
Modified DPMA	6	3.9
Modified ACM	4	2.6
Introductory Database Course Offered in Business School:		
Yes	105	66.0%
No	54	34.0
Introductory Database Course Offered Outside Business School:		
Yes	94	60.6%
No	61	39.4

ADMINISTRATIVE AND ENROLLMENT DATA

The responses to administrative and enrollment queries are tabulated in Table II. The questions in this category cover the number of years the course has been offered, level at which it is offered, credit hours associated with the course, the existence of prerequisites for enrollment in the class, whether or not the course is required for a degree program, the average number of students in each section, and the number of sections taught each academic year.

The introductory database course has been taught over four years in almost half of the schools which offer the class. Sixteen percent were offering the course for the first time during the 1983-84 academic year--an indication of the current recognition of the importance of a background in database for business students.

The course is offered at both undergraduate level and graduate level in 42.9 percent of the responding institutions. Sixteen percent indicated that the introductory database course is offered only at the graduate level, and 41.0 percent noted that the course is taught only at the undergraduate level.

Three hours credit is associated with the course in most schools--81.9 percent. Before allowing students to enroll in introductory database, the vast majority of institutions (97.1 percent) require completion of at least one prerequisite course. Introductory database is required in a degree program in 74.0 percent of the responding institutions which offer the course.

The most frequently noted class sizes were 16-30 and 31-45 students (42.9 percent each). During each academic year, 38.1 percent offered two sections, 19 percent offered only one section, and the remaining 42.8 percent offered from three to six introductory database sections.

Table II

ADMINISTRATIVE AND ENROLLMENT DATA

	<u>Number of Responses</u>	<u>% of Responses</u>
Number of Years Course Offered:		
Over 4 years	50	47.6%
3-4 years	23	21.9
1-2 years	15	14.3
Less than 1 year	17	16.2
Level at which Course is Offered:		
Undergraduate only	43	41.0%
Graduate Only	17	16.2
Both graduate and undergraduate	45	42.9
Credit Hours:		
1 credit hours	4	3.8%
3 credit hours	3	2.9
2 credit hours	86	81.9
4 credit hours	7	6.7
Other	5	4.8
Prerequisite Course Required:		
Yes	101	97.1%
No	3	2.9
Database Course Required in a Degree Program:		
Yes	77	74.0%
No	27	26.0
Average Section Enrollment Size:		
15 students or less	5	4.8%
16-30 students	45	42.9
31-45 students	45	42.9
46-60 students	7	6.7
Over 60 students	3	2.9
Number of Sections Offered each Academic Year:		
1 section	20	19.0%
2 sections	40	38.1
3 sections	14	13.3
4 sections	14	13.3
5 sections	5	4.8
6 or more sections	12	11.4

HARDWARE AND SOFTWARE USED

Table III contains responses to questions on computer hardware and software used in introductory database management courses.

In 30.1 percent of the institutions responding, only mainframes were used for project work. In 24.0 percent, only smaller computers were used. Thirty-seven percent used both mainframes and smaller computers, and 8.7 percent used no computers.

There were 44 different computer makes and models listed as being used in an introductory database course by the responding institutions. The most frequently noted makes and models (five times or more) are shown in Table III. The IBM PC microcomputer is the most frequently used computer--used by 31 of the responding institutions. The computer in second place is the VAX 11/780, followed by IBM's 3081 and UNIVAC's 1100. Another IBM model, the 4341, and the DEC 10 are each used by five institutions for database course projects.

In answer to a query on database management systems used in connection with the introductory database course, 51 different packages were listed. Because many of the database management systems were noted just a few times, only those used by five or more institutions are listed.

The most frequently listed database management system is dBASE II. Other frequently noted database management systems are: IMS, ANALYST and QSORT, TOTAL, INGRES, IDMS, and RIM, SYSTEM 2000, IMAGE, KNOWLEDGEMAN, MODEL 204, and RBASE.

It is apparent from studying the data in Table III that when the IBM PC appeared on the market, it was well received as an educational tool by schools with introductory database courses. It is also interesting that the database management system used by more institutions than any other is dBASE II, a package used on computers classified as minis or micros.

Table III

HARDWARE AND SOFTWARE USED
IN INTRODUCTORY DATABASE COURSES

	<u>Number of Responses</u>	<u>% of Responses</u>
Sizes of Computers Used:		
Mainframes only	31	30.1%
Mini/micros only	24	23.3
Both mainframes and mini/micros	39	37.9
No computers used	9	8.7
Computers Used in 5 or More Institutions:		
IBM PC	31	
DEC VAX 11/780	13	
IBM 3081	8	
UNIVAC 1100	8	
DEC 10	5	
IBM 4341	5	
Other	61	
Database Management Systems Used in 5 or More Institutions:		
dBASE II	29	
IMS	16	
ANALYST & QSORT	16	
TOTAL	15	
INGRES	10	
IDMS	10	
RIM	9	
SYSTEM 2000	6	
IMAGE	5	
KNOWLEDGEMAN	5	
MODEL 204	5	
RBASE	5	
Other	51	

COURSE CONTENT

The topics recommended for coverage by the DPMA in their model curriculum were listed in the questionnaire. Respondents were requested to indicate the approximate total of class time devoted to each topic. The number of responses and the percentage of total responses for each topic are listed in Table IV.

The topic on which the highest number of class periods are spent is "the relational data model." Twenty-three percent of the respondents noted that seven to eight class periods are devoted to relational data models. The topics, "network data model" and "hierarchical data model," are ranked second and third--22 percent and 14.1 percent, respectively.

The DPMA-recommended topic most frequently excluded in the introductory database course is "storage device characteristics and physical input/output." A total of 16 (17.8 percent) of the responses indicated the topic was not covered. Direct file organization and applied data structures are not covered in 12.0 percent and 11.5 percent, respectively. It should be noted that several respondents indicated that these topics were covered in other courses in their institutions.

Table IV

TOPICS INCLUDED	Number of Responses	% of Response
Overview:		
Not Included	3	3.5%
1-2 Class Periods	69	80.2
3-4 Class Periods	9	10.5
5-6 Class Periods	4	4.7
7-8 Class Periods	1	1.2
Storage Device Characteristics & Physical Input/Output:		
Not Included	16	17.8%
1-2 Class Periods	45	50.0
3-4 Class Periods	20	22.2
5-6 Class Periods	9	10.0
7-8 Class Periods	--	----
Indexed Organized Files:		
Not Included	8	9.2%
1-2 Class Periods	47	54.0
3-4 Class Periods	22	25.3
5-6 Class Periods	9	10.3
7-8 Class Periods	1	1.1
Direct File Organization:		
Not Included	11	12.0%
1-2 Class Periods	48	52.2
3-4 Class Periods	20	21.7
5-6 Class Periods	11	12.0
7-8 Class Periods	2	2.2
Applied Data Structures:		
Not Included	10	11.5%
1-2 Class Periods	27	31.0
3-4 Class Periods	31	35.6
5-6 Class Periods	13	14.9
7-8 Class Periods	6	6.9
Data Model Overview:		
Not Included	2	2.2%
1-2 Class Periods	37	40.2
3-4 Class Periods	35	38.0
5-6 Class Periods	13	14.1
7-8 Class Periods	5	5.4
Hierarchical Data Model:		
Not Included	4	4.3%
1-2 Class Periods	38	41.3
3-4 Class Periods	22	23.9
5-6 Class Periods	15	16.3
7-8 Class Periods	13	14.1

Table IV (Continued)

TOPICS INCLUDED	Number of Responses	% of Response
Network Data Model:		
Not Included	3	3.2%
1-2 Class Periods	24	25.5
3-4 Class Periods	22	23.4
5-6 Class Periods	24	25.5
7-8 Class Periods	21	22.3
Relational Data Model:		
Not Included	--	----
1-2 Class Periods	19	20.4%
3-4 Class Periods	18	19.4
5-6 Class Periods	34	36.6
7-8 Class Periods	22	23.7
Database Administration:		
Not Included	7	8.2%
1-2 Class Periods	35	41.2
3-4 Class Periods	19	22.4
5-6 Class Periods	17	20.8
7-8 Class Periods	7	8.2
Other:		
Not Included	1	2.6%
1-2 Class Periods	13	34.2
3-4 Class Periods	12	31.6
5-6 Class Periods	4	10.5
7-8 Class Periods	8	21.1

Table V contains the responses to a query on the textbook(s) used. A total of fifteen database texts were listed by respondents. Responses to this inquiry totalled 115--some institutions noted more than one text. Only those texts which were indicated by more than one respondent are listed.

Kroenke's text, DATABASE PROCESSING, is the text used by more institutions than any other. Date's INTRODUCTION TO DATABASE SYSTEMS ranked second.

Table V

TEXTBOOKS	Number of Responses
Kroenke: DATABASE PROCESSING	53
Date: AN INTRODUCTION TO DATABASE SYSTEMS	16
Martin: MANAGING THE DATA-BASE ENVIRONMENT	8
Bradley: FILE AND DATA BASE TECHNIQUES	8
Bradley: INTRODUCTION TO DATA BASE MANAGEMENT IN BUSINESS	8
Caredenas: DATABASE MANAGEMENT SYSTEMS	6
Atre: DATA BASE: STRUCTURAL TECHNIQUES FOR DESIGN PERFORMANCE AND MANAGEMENT	4
Conen: DATABASE MANAGEMENT SYSTEMS	3
Tsichritzis: DATABASE MODELS	2
Lochovsky:	

TEACHING FACULTY

The results of the inquiry asking for professional and educational data on faculty teaching introductory database courses are listed in Table VI. A total of 180 faculty members teaching database courses in schools of business were reported. Of those, 81.7 percent held doctorate degrees as their highest degree, 16.7 percent have master's, and 1.7 percent have bachelor's degrees. Seventy-seven percent have had database related work experience.

CONCLUSIONS

This survey has provided information on introductory database courses offered in AACSB-accredited schools which may be of value to institutions now offering the course and those planning to offer the course in the future. The results of this research may be interesting not only to AACSB-accredited institutions but also to business schools without the accreditation and to non-business schools.

Table VI

TEACHING FACULTY

	<u>Number of Responses</u>	<u>% of Response</u>
Highest Degree:		
B.S.	3	1.7%
M.S.	30	16.7
Doctorate	147	81.7
Work Experience:		
Yes	138	77.2%
No	42	22.7

REFERENCES

1. Adams, D.R. and Thomas H. Atney (eds), "DPMA Model Curriculum for Undergraduate Computer Information Systems Education," Park Ridge, Illinois: Data Processing Management Association, 1981.
2. Aulgur, Jeretta H., and Joan K. Pierson, "Business School Graduates--How Knowledgeable are They about Computers and Information Systems?" ON-LINE (Spring, 1983), pp. 13-15.
3. Couger, J.D. (ed.), "Curriculum Recommendations for Undergraduate Programs in Information Systems," Communications of the ACM (December 1973), pp. 727-749.
4. Frand, Jason L., "First Annual UCLA Computing Survey of North American Business Schools," Graduate School of Management, University of California at Los Angeles, Los Angeles, CA (June, 1984).
5. Horn, Jeretta, Joan K. Pierson and G. Daryl Nord, "COBOL in Introductory Courses as AACSB-Accredited Institutions--A Profile," ON-LINE (Spring, 1984), pp. 1-5.
6. Nunamaker, Jr., Jay F., J. Daniel Couger, and Gordon B. Davis, (eds.), "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs," Communications of the ACM, Vol. 25, No. 11 (November, 1982), pp. 781-805.

THE DPMA MODEL CURRICULUM
AND AACSB ACCREDITATION

ALDEN C. LORENTS

NORTHERN ARIZONA UNIVERSITY
COLLEGE OF BUSINESS
15066
FLAGSTAFF AZ 86011

ABSTRACT

Many professional programs are often operating within multiple guidelines, standards, and constraints. The DPMA model curriculum is designed to be adopted by colleges of business who are accredited by AACSB (American Assembly of Collegiate Schools of Business). This paper compares the guidelines of the DPMA model curriculum with the guidelines for accreditation by AACSB. The comparison examines areas of compatibility and incompatibility. In general the DPMA model curriculum guideline fits within the AACSB guideline. However, there is not a lot of flexibility left for electives. Also, it is necessary for some of the program to be taken at the lower division.

INTRODUCTION

Professional programs often require many courses in the field of study in order to obtain the skills necessary for that profession. Examples include programs in accounting, engineering and health related fields. There is a tendency in these programs to come up with more requirements than can fit in a normal four year program. There is also a danger that the program may start to deviate from general acceptable guidelines outlined by the accrediting body. The DPMA model curriculum is a professional program often leading to a business degree. It's a program that is, and will be offered by many AACSB schools.

This paper summarizes the requirements of the DPMA model curriculum and compares these requirements with AACSB accreditation guidelines.

AACSB GENERAL GUIDELINES

The main thrust of the AACSB guidelines is to provide a broad education. No one way is prescribed in carrying out this objective. Colleges have a lot of flexibility in setting up curriculum to meet the guidelines.

AACSB sets the following guidelines:

- (1) An undergraduate school of business should concentrate its professional courses in the last two years of a four-year program, and should offer only a limited amount of work below the junior year.
- (2) The first two years should be primarily foundation work that would normally include courses in communications, mathematics, social sciences, humanities, and the natural sciences.

(3) Sample business courses at the lower division could include principles of accounting, principles of economics, business law and statistics. Sample business courses at the upper division could include principles of finance, management, and marketing.

(4) Normally, forty to sixty percent of the program should be in business and economics and forty to sixty percent outside of business.

(5) The equivalent of at least one year of work should be in courses providing students with a common body of knowledge in business.

The guideline for the common body of knowledge in business is as follows:

- (1) a background of the concepts, processes and institutions in the production and marketing of goods, and the financing of the business enterprise.
- (2) a background of the economic and legal environment along with ethical considerations and social and political influences.
- (3) a basic understanding of the concepts and applications of accounting, of quantitative methods and management information systems, including computer applications.
- (4) a study of organization theory, behavior, and interpersonal communications.
- (5) a study of administrative processes under conditions of uncertainty including integrating analysis and policy determination at the overall management level.

Institutions may implement curricula that varies with the common body of knowledge as long as an overall high quality is maintained.

Given these guidelines, a typical business program with an emphasis or major in some area might be structured as follows:

	SEMESTER HOURS	
	SAMPLE	RANGE
General studies	42	(40-50)
Business core	39	(30-45)
Major core	18	(9-18)
Electives	17	(10-35)
Total semester hours	125	

DPMA MODEL CURRICULUM

The primary objective of the DPMA model curriculum is to provide graduates with the knowledge, abilities, and attitudes to function effectively as applications programmer/analysts.

The model curriculum set forth the following specific objectives in meeting this overall objective.

- (1) to provide an understanding of the goals, functions, and operations of business organizations.
- (2) to provide an understanding of the information needs and the role of information systems in these organizations.
- (3) to provide the analytical and technical skills of identifying, studying, and solving information problems within organizations.
- (4) to provide communications and human relations skills for effective interaction with organization members, especially with the users and developers of information systems.
- (5) to provide knowledge and ability for effective management of information systems projects.
- (6) to instill a professional attitude and seriousness of purpose about Computer Information Systems as a career field.
- (7) to provide the background for further study of and professional advancement in the field of Computer Information Systems.

The CIS model curriculum consists of the following course blocks.

- (1) a set of general education courses in the arts, sciences, and the humanities, to broaden intellectual awareness and to serve in the development of cultural literacy and analytical and evaluative methods.
- (2) a minimum set of business support courses.
- (3) a core of seven (21 semester hours) of CIS courses.
- (4) three (9 semester hours) of elective CIS courses.

Four of the seven core courses are at the lower division. All of the elective courses are at the upper division. There are no guidelines stated regarding non-business electives.

COMPATIBILITY WITH AACSB GUIDELINES

The DPMA model curriculum was developed with most of the AACSB guidelines in mind. The requirements in general studies, the common body of knowledge in business, and the program objectives are very compatible. The following chart illustrates a possible curriculum structure that implements the model curriculum within AACSB guidelines.

CIS PROGRAM COMPONENTS (SEMESTER HOURS)

GENERAL STUDIES (44)

ENGLISH
MATH
SCIENCE
HUMANITIES
SOCIAL SCIENCE

BUSINESS CORE (39)

ACCOUNTING
ECONOMICS
CIS
QUANTITATIVE METHODS
PRODUCTION
LAW
MANAGEMENT
MARKETING
FINANCE
INTEGRATING SEMINAR

CIS CORE (18)

PROGRAM DEVELOPMENT I AND II
SYSTEMS I AND II
DATABASE
SOFTWARE DEVELOPMENT PROJECT

ELECTIVES (9-12)

DECISION SUPPORT
ADVANCED DATABASE
NETWORKS
IS MGT AND PLANNING
AUDIT AND CONTROLS

OUTSIDE BUSINESS ELECTIVES (12 - 15)

COMMUNICATIONS
MATH
COMPUTER SCIENCE
SOCIAL AND BEHAVIORAL SCIENCE

TOTAL HOURS IN BUSINESS	69	55%
TOTAL HOURS OUT OF BUSINESS	56	45%

This curriculum assumes that the introductory CIS course is part of the business core. Other CIS content may be part of the business core in many AACSB schools in the near future.

The model curriculum probably pushes programs closer to the 60/40 limit in most AACSB schools. There will be less room for electives outside the business program. This depends some on the general studies requirements that are normally controlled university wide. In some schools the general studies requirement may eliminate all electives outside of business.

The model curriculum has four of its seven core courses at the lower division. This is somewhat incompatible with the AACSB guideline of taking the majority of the business courses at the upper division. Professional programs of this type, normally require a number of courses that are dependent on each other. Three years are required for the sequence of dependencies. Accounting programs have a similar situation, and some have gone to a five year program.

The flexibility of the AACSB guidelines allows for a non-standard implementation of curriculum content. Instead of having separate courses in each area, course content can be integrated so that multiple requirements are being met at the same time. An example might be to teach a policy-making course with an MIS orientation. Another example would be to teach systems where all the cases and problems are related to a marketing department. Using this approach does not necessarily save time, but may be more effective.

BS IN COMPUTER INFORMATION SYSTEMS

The college of Business at Northern Arizona University has offered a BS in Business with an emphasis in DP/CIS for a number of years. A proposal is currently in process to offer a BS in CIS. The program is structured as follows:

CIS CORE 18 HOURS

PROGRAMMING I II AND III
SYSTEMS ANALYSIS AND DESIGN
DATABASE I AND II

CIS ELECTIVES 12 HOURS

MODELING AND DECISION SYSTEMS
SYSTEMS SOFTWARE
MIS SEMINAR
SYSTEMS DESIGN II
DISTRIBUTED NETWORKS
ADVANCED SYSTEMS SEMINAR

The introductory CIS course and an introductory MIS course are in the business core. MIS is an upper division course that introduces business systems, systems analysis, and decision support systems.

Programming I is set up to teach fundamentals of program design using structured methodology. Programming II and III are the core COBOL courses.

Database I is data structures, relational and fourth generation languages. Database II goes into hierarchical and network databases.

The objective of this program is to provide an alternative for students who want more depth in information systems or computing systems. A separate degree also gives a little more visibility to the program.

The credit hours required still fit within the model presented in the chart above.

CONCLUSION

It appears that the DPMA model curriculum can be implemented very easily within AACSB accredited schools. The objectives are very compatible and the program structure fits within the guidelines very well.

There are some cautions. First, some of the professional courses (programming) are taken at the lower level. This does not appear to be a significant problem. Accounting programs have been in this mode for some time. However, it is an area that should be reviewed in future program revisions. Second, most programs will not have much room for general electives. If a student wants to take other courses, the credits will probably be beyond the graduation requirements.

REFERENCES

1. _____, Accreditation Council Policies, Procedures, and Standards, AACSB, 1983-84.
2. Adams, D. R. (ed.), "Curriculum Recommendations for Undergraduate Programs in Computer Information Systems Education", DPMA Education Foundation, Park Ridge, Il (1981).
3. Couger J. D., (ed.), "Updated Information System Curriculum Guidelines", Computing Newsletter, April 1982.
4. Nunamaker, J. F. (ed.), "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs", Communications of the ACM, November 1982.

CIS CURRICULUM IN A TWO-YEAR PROGRAM

Marvin Gore
Dean, Business Division

Mt. San Antonio College
1100 N. Grand Avenue
Walnut, California 91789

ABSTRACT

The DPMA Model Curriculum for undergraduate computer information systems education was designed primarily for four-year programs. Also, the mission of the two-year, community college differs from that of four-year schools in several respects. Nonetheless, the CIS curriculum lends itself to implementation at two-year institutions. Factors affecting the implementation of the CIS curriculum in community colleges are discussed. Particularly important for transfer students are articulation agreements between the community college and local four-year institutions and post-articulation flexibility.

FACTORS AFFECTING THE TWO-YEAR CIS PROGRAM

The principal focus of the DPMA Model Curriculum for undergraduate computer information systems (CIS) education is the four-year schools; however, as the Model Curriculum quote of Figure 1 indicates, the needs of students transferring from two year, community colleges are recognized.

Three factors that significantly affect the achievement of the goals of the Model Curriculum at the two-year program level are:

1. The Mission of the two-year, community college.
2. The community college CIS curriculum.
3. Articulation requirements/agreements.

MISSION

The mission of the community college is to provide life-long education and career learning opportunities. Emphasis is upon both occupational employment and transfer to a four-year institution. The student population tends to be much less homogenous than that of a four-year college or university, and the needs of this population are diverse. Figure 2 lists the four types of students whose needs are most relevant to a discussion of the CIS curriculum. The characteristics of these student types are:

1. The Job-Skill Seeker: The student who will take one or more courses in order to acquire or improve job skills. This category includes the "jobout", who leaves school for a job before completing all of the courses required for a degree or certificate.
2. The Certificate Seeker: The student who completes a core (at least six and usually more) of courses in order to obtain a certificate that designates a level of competency.
3. The Degree Seeker: The student who completes 60 or more units of study in order to obtain an A.S. or A.A. degree. These units include not only the core of coursework required for a certificate, but also courses that provide additional career-field knowledge or meet the general education and other graduation requirements of the college. This person may or may not be a "transfer" student.
4. The Transfer Student: The student who completes considerable coursework at a community college, but whose short-term educational goal is a degree from a four-year college or university. This student may or may not complete a certificate or two-year program.

Although the first three categories of students might be considered "non-transfer", many of these students do resume their education at a later date. Fortunately, many of their immediate and future needs can be served by courses that conform to the recommended lower-division CIS offerings. A complicating factor is the need to provide terminal two-year students with a significant amount of course content that is designated as upper-division in the four-year CIS curriculum. Examples are computer system hardware and software architecture, data base management systems, and the complete system development life cycle. Other courses, such as BASIC programming and microcomputer selection, are offered

to meet identified community vocational needs and are not reflected in the current CIS Model Curriculum.

CURRICULUM

A determined effort has been made at Mt. San Antonio College to establish certificate and degree programs that can meet the vocational needs of our community and, yet, remain compatible with the objectives and content of the CIS Model Curriculum. We feel that we have been successful.

We do not have separate tracks for "transfer" and "non-transfer" students. As previously pointed out, CIS recommended courses often meet the needs of all four types of CIS students. Other reasons are:

1. Many students do not decide upon CIS as a major until they have completed a course or two in the subject area.
2. There are no reliable means of labeling entering CIS students as transfer or non-transfer.
3. Students who do not continue their education upon leaving a two-year school may do so at a future time.

The Mt. San Antonio College certificate and degree programs are offered by the Computer Information Systems (CIS) Department. They are discussed below.

Certificate Program

Figure 3 lists the required courses and restricted electives needed to obtain the Certificate in Computer Programming. The Computer Programming Certificate requires the completion of over 30 units of coursework. It consists of the core of CIS courses required for the A.S. Degree in Data Processing, including those designed to meet the Model Curriculum recommendations for lower-division students. Actually, the number of units required is somewhat greater than indicated because all of the programming classes have laboratory sections that generate additional units of credit.

Degree Program

Figure 4 lists the required lower-division and upper-division courses in the DPMA Model Curriculum for CIS, and Figure 5 lists the elective courses. Figure 6 shows the Mt. San Antonio CIS course requirements for the A.S. Degree in Data Processing. The corresponding lower-division Model Curriculum courses are shown in parenthesis, (). The upper-division Model Curriculum courses with significant related content are indicated by brackets, [].

The CIS degree program is designed to serve transfer students and to provide entry-level employment for students who do not choose to continue their formal education at this time in their careers. Although programming is a strong program component, there is a major emphasis on systems analysis and design.

Data 20, Systems Analysis, and Data 48, Seminar, which is a capstone student project, provide a sequence of studies and activities that cover the structured analysis, design, and development of computer related business information systems. We feel that a knowledge of all of the phases of the system development life cycle is essential for two-year data processing and business majors because they will encounter many computer-related business information systems at work.

For this reason we require both the introductory data processing and systems analysis courses for many degree programs in business in addition to the Computer Information Systems program. Examples are: Banking and Finance, Accounting, Bank Operations, and Business Management.

There are college requirements for granting the A.S. Degree to CIS majors that add to the technical subject matter. These are (1) competencies in mathematics and in reading, writing, and speaking. (2) health and physical well-being, and (3) general education. A mandatory assessment test in English is administered before students are admitted to introductory level classes in data processing.

The general education requirement is for a minimum of 18 units, distributed as follows:

1. Natural Sciences (3 Units)
2. Social and Behavioral Sciences (6 Units)
3. Humanities (3 Units)
4. Language and Rationality (6 Units)

Fortunately, because of the number of CIS units required for the major, six of the required general education units are met by courses in the major. Data 10A, Data Processing, meets three units of the social and behavioral science requirement, and any programming language accounts for three units of the language and rationality requirement.

Facilities

In spite of limited funds, the Business Division has been fortunate over the past several years in having had strong administrative support for the development of an instructional computer center (ICC) dedicated to the needs of CIS and other business students. The principal resources in this center, which occupies approximately 3,000 square feet of floor space are: An IBM System 34, a remote job entry link and terminals directly attached to a dedicated IBM 4341, and Apple II and IBM microcomputers. A description of the Instructional Computing Center appears as Figure 7. Starting with Data 10A, students are provided with hands-on experience on all of the computers in the ICC.

Future Trends

There will be an increasing emphasis upon data base management and distributed data processing, including both large system and microcomputer applications. A strong emphasis upon structured methods for systems analysis and design will continue, with applications in all advanced programming courses. More micro-oriented courses will be offered.

A microcomputer classroom (MCC) of IBM personal computers became operational in September 1985. The MCC layout is shown in Figure 8. It facilitates teaching personal computer applications, such as word processing, spread sheets, and integrated software packages. Microcomputer Software Applications (Data 12) and Relational DataBase Management for Microcomputers (Data 37) are examples of courses that are taught in this classroom. The MCC also provides a facility for instructors to teach microcomputer applications in non-CIS disciplines, such as accounting, marketing, economics, and small business management.

One significant effect of technological advances will be the broadening of the CIS curriculum in both two-year and four-year schools. This suggests the need for options, or, at least, the careful counseling of students into related sequences of courses.

ARTICULATION

University A and University B

Effective articulation with the four-year universities to which community college students are most likely to graduate is essential. To a large degree this can be achieved through mutual, cooperative efforts. Figure 9 shows the current status of our articulation agreements with the two universities at which most of our students continue their education--either as transfer students or at a later time. We have designated these as University A and University B.

Figure 9 is the basis for two important observations. The first is that community colleges often must deal with the guidelines of the DPMA Model Curriculum as implemented (or not implemented) locally by more than one four-year school. Actually, University A was very active in the development of the Model Curriculum, and its curriculum emphasis is best described as Computer Information Systems (CIS). The emphasis of the curriculum of University B is best described as Management Information Systems (MIS).

The second observation is that articulation occurs with both universities, however it is more effective with the CIS-oriented school. Neither University A nor University B offers a course in BASIC, which is less of a standard for business programming than is COBOL. Nonetheless, University B includes a strong exposure to BASIC in MAN SCI 265, Information Systems and Programming. For this reason both Data 10A, Data Processing, and Data 20, BASIC, are required as conditions of articulation. This is not a problem, since BASIC is a strong and popular component of our community-oriented program.

University A accepts Data 20, Systems Analysis plus Data 48, Seminar, for CIS-4 and Data 20 plus Data 26, COBOL Programming, for its accounting program. Generally, a community college systems analysis course is difficult to articulate directly to one of the series of systems courses (lower and upper division) defined in the Model Curriculum. One reason for this is provided by the authors of the DPMA Model Curriculum. They point out, in their discussion of CIS-4, that community colleges may need to "tailor topics to suit local needs and increase the ability of the course to stand alone." This means that some of the content of CIS-5, which is an upper-division course, is included. Another reason is that the distribution of emphasis is among topics such as system flowcharts, dataflow diagrams, and prototyping may be different.

Post-Articulation

At the community college we are finding that the number of students arriving with many of the skills that we teach is increasing. For example, in secondary schools and in vocational schools many students have acquired knowledge and skills equivalent to those taught in some of our courses. Although we cannot provide college credit for the courses that they have taken, we do provide opportunities for waiver-by-petition or for credit-by-examination. If a student takes an examination, typically in a programming language, the grade earned is awarded. However, credit-by-examination implementation is a post-articulation event because the credit earned is not granted to the student until twelve units of college-level work have been completed on our campus.

Also, we plan to provide a short-course opportunity for students who enter our program with college credit from another institution, who have earned credit-by-examination, or who have received credit for a TV course in order to familiarize them with the ICC laboratory equipment.

Extension of the post-articulation concept to the CIS relationships between four-year and two-year schools can be productive. An example is waiver-by-petition for a programming course, such as assembly language, which is taught at a community college, but offered as an upper-division course at a four-year university. This process can be extended through waiver-by-petition or credit-by-examination, to include courses now appearing as upper-division CIS offerings and also becoming increasingly prominent in community college curricula.

At a time when many of our high-tech programs are impacted and teachers and facilities are scarce resources, there is a need for the four-year and two-year schools to work together closely to the benefit of all deserving students. I am certain that this paper has touched only briefly upon the cooperative and creative approaches that are possible.

References:

1. DPMA Model Curriculum for Undergraduate Computer Information System, prepared by the DPMA Education Foundation Committee on Curriculum Development, Park Ridge, Illinois, 1981
2. Mt. San Antonio College Bulletin, Mt. San Antonio College, Walnut, California, 1984/85

Quote from preface to CIS Model Curriculum:

"These curriculum guidelines are designed primarily for four-year undergraduate programs...The recommendations also have implications for two-year community college curriculums in data processing where transfer credit towards CIS programs is offered. These guidelines recognize the need for articulation between two and four-year programs, as well as the value of having terminal degree courses of study at both levels."

FIGURE 1. FOCUS OF THE DPMA MODEL CURRICULUM FOR UNDER-GRADUATE COMPUTER INFORMATION SYSTEMS EDUCATION

- JOB-SKILL SEEKER
- CERTIFICATE SEEKER
- DEGREE SEEKER
- TRANSFER STUDENT

FIGURE 2. STUDENT TYPES RELEVANT TO CIS CURRICULUM

COMPUTER PROGRAMMING
CERTIFICATE

Required Courses (19 units)

DATA 10A	Data Processing*	3
DATA 10B	Data Processing	3
DATA 20	Systems Analysis	3
DATA 26	COBOL Programming*	3
DATA 48	Data Processing Seminar*	3
BUSA 7	Principles of Accounting	4

Restricted Electives (select 15 units
from the following:)

DATA 12	Microcomputer Software Applications	3
DATA 28	BASIC Programming*	3
DATA 34	RPG II Programming*	3
DATA 35	Data Base Management*	3
DATA 36	Advanced COBOL Programming*	3
DATA 37	Relational Database Manage- ment for Microcomputers	3
DATA 38	Advanced BASIC Programming*	3
DATA 44	Advanced RPG II Program- ming *	3
DATA 46A	Assembler Language Coding*	3
DATA 46B	Assembler Language Coding*	3
BUSA 8	Principles of Accounting	4

*Requires concurrent enrollment in a laboratory course. Hours for the laboratory course will be arranged by the student at the student's convenience.

FIGURE 3. CERTIFICATE PROGRAM IN COMPUTER PROGRAMMING

CIS Core Courses

The curriculum core is comprised of the following four lower-division (freshman/sophomore level) courses and three upper-division (junior/senior level) courses.

- CIS-1 INTRODUCTION TO COMPUTER-BASED SYSTEMS.
An introduction to computers and data processing taught as a general education course for all students (lower division).
- CIS-2 APPLICATIONS PROGRAM DEVELOPMENT I.
A beginning computer problem solving and programming course using COBOL as the vehicle language (lower division).
- CIS-3 APPLICATIONS PROGRAM DEVELOPMENT II.
An advanced computer problem solving and programming course using COBOL (lower division).
- CIS-4 SYSTEMS ANALYSIS METHODS.
An overview of the systems development life cycle with emphasis on techniques and tools of system documentation and logical system specifications (lower division).
- CIS-5 STRUCTURED SYSTEMS ANALYSIS AND DESIGN.
Advanced coverage of the strategies and techniques of structured systems development (upper division).
- CIS-6 DATABASE PROGRAM DEVELOPMENT.
A course emphasizing software design and programming in a database environment (upper division).
- CIS-7 APPLIED SOFTWARE DEVELOPMENT PROJECT.
A capstone systems course integrating the knowledge and abilities gained through the other computer-related courses in the curriculum within a comprehensive system development project (upper division).

FIGURE 4. CIS CORE CURRICULUM

CIS Elective Courses

In addition to the seven core courses, three upper-division elective courses are required. These electives should be chosen from the following set of courses:

- CIS-8 SOFTWARE AND HARDWARE CONCEPTS.
A survey of technical topics related to computer systems with emphasis on the relationships between hardware architecture, systems software, and applications software.
- CIS-9 OFFICE AUTOMATION.
An examination of the office as a center of business activity, operational logistics, and decision support, and the impact of automation on the office environment.
- CIS-10 DECISION SUPPORT SYSTEMS.
An analysis of the highest level of information support systems which aid the manager in the decision-making process.
- CIS-11 ADVANCED DATABASE CONCEPTS.
An in-depth investigation of data modeling, system development, and data administration in a database environment.
- CIS-12 DISTRIBUTED DATA PROCESSING.
An examination of the features and impact of distributed systems in the business enterprise.
- CIS-13 EDP AUDIT AND CONTROLS.
An introduction to EDP auditing with emphasis on EDP controls, audit types, and audit techniques and their effects on system development.
- CIS-14 INFORMATION SYSTEMS PLANNING.
An introduction to the financial, technical, and strategic information systems planning process.
- CIS-15 INFORMATION RESOURCE MANAGEMENT.
A seminar in information systems management with emphasis on planning, organizing, and controlling user services and managing the systems development process.

FIGURE 5. CIS ELECTIVE COURSES

DATA PROCESSING
ASSOCIATE IN SCIENCE DEGREE

Required Courses (19 units)

DATA 10A	Data Processing*	3-(CIS-1)
DATA 10B	Data Processing	3-[CIS-8]
DATA 20	Systems Analysis	3-(CIS-4)
DATA 26	COBOL Programming*	3-(CIS-2)
DATA 48	Data Processing Seminar*	3-[CIS-5]
BUSA 7	Principles of Accounting	4

Restricted Electives (select 15 units from the following:)

DATA 12	Microcomputer Software Applications	3
DATA 28	BASIC Programming*	3
DATA 34	RPG II Programming*	3
DATA 35	Data Base Management*	3
DATA 36	Advanced COBOL Programming*	3-(CIS-3)
DATA 37	Relational Database Management for Microcomputers	3-[CIS-6]
DATA 38	Advanced BASIC Programming*	3
DATA 44	Advanced RPG II Programming*	3
DATA 46A	Assembler Language Coding*	3
DATA 46B	Assembler Language Coding*	3
BUSA 8	Principles of Accounting	4

- () Lower Division Model Curriculum Courses
[] Upper Division Model Curriculum Courses

*Required concurrent enrollment in a laboratory course. Hours for the laboratory course will be arranged by the student at the student's convenience.

FIGURE 6. DEGREE PROGRAM IN DATA PROCESSING AND COMPARABLE DPMA CIS COURSES

THE FOLLOWING FACILITIES ARE AVAILABLE TO DATA PROCESSING STUDENTS
AT MT. SAC:

INSTRUCTIONAL COMPUTER CENTER

I.B.M. SYSTEM 34

28 TERMINALS
FULL PAGE EDITOR - VIM
SINGLE LINE EDITOR - SEU

256K OF MAIN STORAGE AVAILABLE FOR PROCESSING

I.B.M. PRINTER - 600 LINES PER MINUTE
(for programs processed on SYSTEM 34)

THE SYSTEM 34 IS CURRENTLY USED TO ENTER AND PROCESS 10A
ASSIGNMENTS, RPG PROGRAMS, ADVANCED COBOL PROGRAMS.

THE SYSTEM 34 IS ALSO USED TO ENTER AND TRANSFER COBOL
PROGRAMS TO AN I.B.M. 4331 COMPUTER ACROSS CAMPUS FOR
PROCESSING. THIS IS DONE THROUGH A REMOTE JOB ENTRY
TERMINAL.

I.B.M. 4331 - INSTRUCTIONAL

8 TERMINALS
FULL PAGE EDITOR - XEDIT
2 MB OF MAIN STORAGE AVAILABLE FOR PROCESSING

I.B.M. PRINTER - 600 LINES PER MINUTE
(for programs processed by I.B.M. 4331)

THESE TERMINALS ARE CONNECTED DIRECTLY TO THE 4331 BY COAXIAL
CABLE AND ARE USED TO ENTER AND PROCESS COBOL PROGRAMS,
ASSEMBLER PROGRAMS, DATA BASE ASSIGNMENTS.

MICRO COMPUTERS

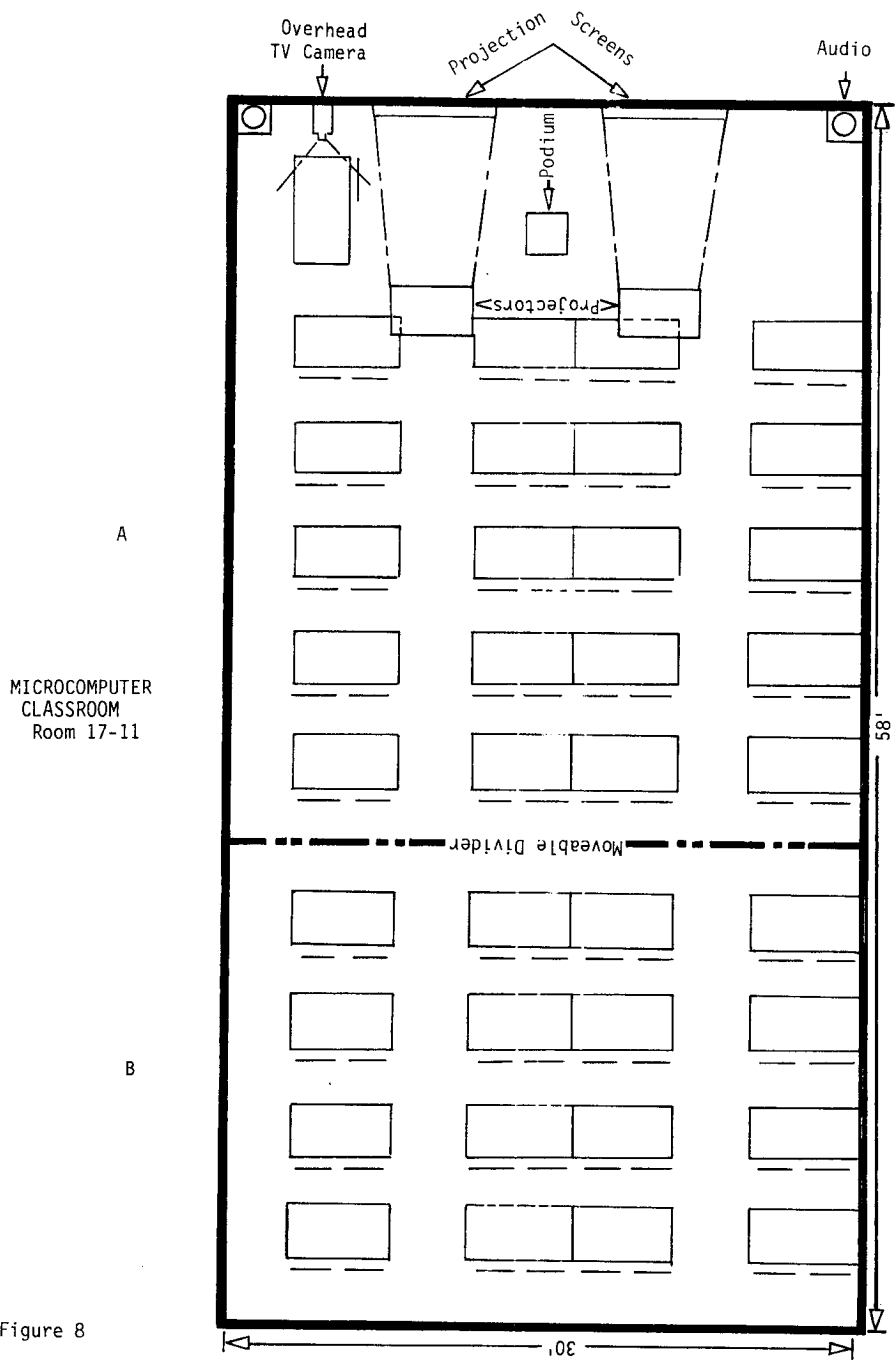
24 APPLE IIes AND 15 I.B.M. PC's

25 PRINTERS

THE MICROCOMPUTERS ARE SET UP TO RUN ON MICROSOFT CP/M OPERATING
SYSTEM AND USE MBASIC.

THE MICROCOMPUTERS ARE CURRENTLY BEING USED TO ENTER AND PROCESS
BASIC PROGRAMS AND DATA BASE ASSIGNMENTS.

FIGURE 7. INSTRUCTIONAL COMPUTER CENTER



MICROCOMPUTER
CLASSROOM
Room 17-11

Figure 8

MT. SAN ANTONIO COLLEGE	UNIVERSITY A (CIS Curriculum)	UNIVERSITY B (MIS Curriculum)
DATA 10A - Data Processing	CIS 110 (CIS-1)	MAN SCI - 265 (1)
DATA 20 - Systems Analysis	(2)	(3)
DATA 26 - COBOL	CIS 144 (CIS-2)	
DATA 36 - Advanced COBOL	CIS 244 (CIS-3)	MAN SCI 270 (4)

- (1) DATA 10A + DATA 20 (BASIC) required
- (2) DATA 20 + DATA 48 required for CIS majors
DATA 20 + DATA 26 required for accounting majors
- (3) Systems Analysis-only an upper division course, petition possible.
- (4) DATA 26 + DATA 36 required.

FIGURE 9. ARTICULATION WITH LOCAL FOUR-YEAR SCHOOLS

CIS CURRICULUM - THE CANADIAN ACADEMIC PERSPECTIVE
MARIA A. FINE
LAKEHEAD UNIVERSITY
THUNDER BAY, ONTARIO, CANADA

ABSTRACT

This paper reports the results of a survey of Canadian academic opinion about the Data Processing Management Association's model curriculum and accreditation in the information systems discipline. The purpose of the research was twofold. Firstly questions were asked to determine how familiar academics were with the DPMA and the model curriculum, how extensively it was being followed and their opinion of it. Secondly, academics were asked to respond to questions on accreditation. Specific questions were asked concerning the need for accreditation and how it should be administered.

INTRODUCTION

Computerized information systems have become increasingly important in the effective management of organizations. Universities and colleges have developed a variety of undergraduate programs to meet the needs of information system users in business. Thus the composition of curriculum and the related issue of accreditation in the information systems field continue to be of interest to both academics and the business community.

Several professional organizations, most notably the Association for Computing Machinery (ACM) and the Data Processing Management Association (DPMA), have been active in curriculum design. The DPMA's Education Foundation Committee on Curriculum Development has produced, in their 1981 report entitled "DPMA Model Curriculum for Undergraduate Computer Information Systems Education", curriculum guidelines for the computer information systems discipline.

This paper reports the results of a survey of Canadian academics about the DPMA Education Foundation's model curriculum. In addition, information was gathered about accreditation. The purpose of the survey was twofold. Firstly, questions were asked to determine how familiar academics were with the DPMA and the model curriculum, how extensively the model curriculum guidelines were being used and the respondent's opinion of the curriculum document. Secondly, questions were asked concerning accreditation in the information processing field; specifically if there was a need for accreditation and how it should be administered.

METHODOLOGY

A survey package was mailed out to deans of business schools and/or computer science departments at 165 colleges and universities across Canada. The package included copies of the DPMA report entitled "The DPMA Model Curriculum for Undergraduate Computer Information Systems Education", a DPMA pamphlet entitled "Computer Information Systems Curriculum" and the survey questionnaire. The deans were requested to direct the package to an appropriate faculty member. Of the 165 institutions surveyed, 50 usable responses were received for an overall usable response rate of 30%.

LIMITATIONS

It should be noted that the responses are those of individual faculty members at various colleges and universities. Therefore, these responses do not necessarily reflect the policies of the institutions or the general perceptions of the

Canadian academic community.

SURVEY RESULTS

The survey instrument consisted of three sections. The first section contained general questions about the type of institution and programs offered. Section two contained questions related to DPMA and the model curriculum. Section three was concerned with accreditation.

SECTION I - GENERAL RESULTS

From the 50 responses received the following characteristics are noted:

- (1) the response rate from universities (24 of 49 universities surveyed) was significantly higher than the community college response rate (26 of 116).
- (2) the two most common programs offered by responding institutions were a degree program in business (34%) and a diploma in data processing (18%).

Other programs offered included degree programs in computer science and diploma programs in business. It should be noted that universities in Canada are generally degree granting institutions while, community colleges offer diploma programs.

SECTION II - DPMA/MODEL CURRICULUM RESULTS

When respondents were asked if they were aware of DPMA Canada ACFOR prior to receiving the survey, only 44% stated that they were yet 67% stated that they were aware of the model curriculum. This apparent discrepancy may be attributed to the question format itself. ACFOR is the French acronym for DPMA and it is quite possible that the English speaking respondents were unaware of DPMA Canada ACFOR's official name.

As Table 1 indicates the majority of respondents stated that they did not follow the model curriculum guidelines. However, Table 2 indicates a large number of respondents felt their institutions existing curriculum covered over 75% of the models core courses. When these results are combined with those of Table 3 almost 50% of all respondents felt their institutions covered over 75% of the model curriculum's core courses.

Table 1

Does your institution follow the DPMA model curriculum presently?

	Yes	No
University	16.7%	83.3%
College	16.7%	83.3%
Total	16.7%	83.3%

Table 2

If your institution does not presently follow the DPMA model curriculum, what percentage of core courses would you say are covered by your existing curriculum?

	1-25%	26-50%	51-75%	76-100%
University	47.1%	23.5%	11.8%	17.6%
College	9.5%	4.8%	14.3%	71.4%
Total	26.3%	13.2%	13.2%	47.4%

Table 3

If your institution does follow the model curriculum guidelines how extensively is it covered?

	26-50%	51-75%	76-100%
University			100%
College	25.0%	25.0%	50%
Total	12.5%	12.5%	75%

When respondents were asked whether their institutions would alter their present curriculum to meet the requirements of the model, a majority (60%) responded negatively. Respondents were asked for reasons why they felt their department would not alter their existing curriculum. Although this question was open-ended two categories of responses emerged from the data as significant. As Table 4 indicates approximately 32% of academic responses fell into category 3 entitled "Developed own curriculum based on local needs or government mandate". The second category of importance, category 4 was entitled "Lack of resources or no data processing option available". Twenty-five percent of the respondents chose this category.

Table 4

Why wouldn't your institution consider altering its model curriculum requirements?

	1 Using It Now	2 Possible In Future	3 Locally or Prov. Gov't.	4 No DP Option Lack Resources	5 Other
University	15.0%	10.0%	10.0%	40.0%	25.0%
College	16.7%	12.5%	50.0%	12.5%	8.3%
Total	15.9%	11.4%	31.8%	25.0%	15.9%

Respondents were asked for their opinion of the model curriculum. Eighty percent of those responding to the question had a favorable opinion of the curriculum guidelines. A majority of those who had not previously seen the curriculum guidelines were in this group. Thus, it would appear that there is support for the model curriculum guidelines among both users and nonusers.

With respect to this section there appears to be a statistically significant difference in responses from universities versus colleges. A larger

percentage of college than university respondents were familiar with the DPMA organization. Table 2 indicates that a significantly higher proportion of college versus university academics felt that although their institutions did not follow the model curriculum guidelines they covered over 75% of its content. Table 4 appears to indicate a difference in opinion between university and college respondents with respect to why curriculums would not be altered by their institutions. College respondents favored category 3, while university respondents favored category 4.

SECTION III - ACCREDITATION RESULTS

Table 5 indicates a large percentage of the respondents felt that there was a need for professional accreditation in the computer information systems field. However, although respondents indicated a need for accreditation, answers varied widely on the form accreditation should take. Opinions given included: accredit the academic program, not the individual; make the accreditation process comparable to other professional designations; model it after the CCP/CDP; grant a degree or diploma; and consider international accreditation.

Table 5

Do you feel that there is a need for professional accreditation in the computer information systems field?

	Yes	No
University	73.9%	26.1%
College	66.7%	33.3%
Total	70.2%	29.8%

The majority of respondents were in favor of a standardized exam being used for the accreditation process. Respondents also favored a period of work experience before a standardized exam was administered. With respect to educational standardization, 62% felt there was a need for it. However, as with the need for accreditation, opinions varied on what standardization implied.

SUMMARY

Respondents had a favorable opinion of the model curriculum guidelines.

Generally, institutions in Canada did not follow the curriculum guidelines however, a significant number did state that they covered much of what is contained in the core courses of the guidelines.

The results seem to indicate a greater coverage of the curriculum courses by colleges rather than universities. University respondents (largely business department faculty) indicated that the lack of resources was a major obstacle to implementation of the model curriculum.

The need for some form of accreditation in information systems was expressed by a majority of respondents. However, there was no consensus on either the administration of the process or the form of standardization.

THE UW-WHITEWATER MANAGEMENT COMPUTER SYSTEMS PROGRAM:
A CURRICULUM ADAPTED TO CHANGE

Jacob Gerlach, Iza Goroff, and Robert Horton
University of Wisconsin-Whitewater
Whitewater, WI 53190

Change is a permanent characteristic of the field of information systems. Universities are not normally organized to accept change readily. The curricular process usually involves a number of steps, typically including at least:

- . recognition of the need for change
- . initiation by an individual or small group
- . approval by department
- . approval by "school" or "college" curriculum committee
- . approval by university curriculum committee
- . approval by an academic administrator
- . publication of a catalog.

If the catalog is published annually the process can take over a year and if the catalog is published biannually as it is at UW-WHITEWATER the whole process can easily take two years. Because this time lag is not acceptable we have attempted to institutionalize change into our program.

MONITORING CHANGE

To understand the effects of change it is not enough to read the scholarly and industrial periodicals; publicity campaigns or theoretical advances frequently have little or no permanent effect on the field of information systems as it is practiced. Since its inception the major has relied upon the MCS Executive Advisory Board as one of its chief resources to evaluate the impact and relevance of new developments. This board, which now consists of twenty-two data processing professionals chosen from among the regions' industrial firms, is charged with providing guidance for the faculty in establishing proper goals and directions for the MCS Program and insuring that the best of industry practices and standards are brought to the classroom. The input from the Advisory Board is then carefully weighed with pedagogy considerations and published curriculum recommendations by DPMA (1) and ACM (6).

Recently our industry representatives recommended two curricular changes:

- . requiring a higher percentage of on-line assignments, earlier in the program
- . understanding the role and availability of "end user" software.

The on-line assignments were relatively easy to place within those courses devoted to the concepts of programming. The "end user" software change has resulted in a new course. The research into the design of that course is the subject of another paper (5). The addition of a new course to a curriculum frequently has severe consequences, especially for a major which already has a maximum number of required courses; to add a course means to eliminate another course. We proceeded with a major restructuring of several courses weeding out topics that were dated or of lesser value, shifting and working the remaining topics into existing courses, and saving all of the important "odds and ends" to be included with the "end user" concepts in the new course TOPICS IN COMPUTER APPLICATIONS.

The obstacles to change are an inflexible curricular structure and the existence of detailed course descriptions. Major changes in curriculum always cause "institutional headaches". Administratively, what must be done if a course, required by the current catalog, is deleted? How long must the faculty wait before the revised courses can first be taught? Will the "hot" topic still be "hot" when the change is finally in place and the course first taught?

The course, Topics in Computer Applications, is designed to alleviate most of these difficulties. Courses with similar titles have been given at many institutions with the course an elective and the content chosen by the instructor. That is not the case here; the course is required, and the topics are chosen by the faculty as a whole based on industry input. If necessary this course could cover completely different topics from a totally new point of view without any need for curricular

action. More standard courses, where changes are slower and smaller, still use the traditional course titles and course descriptions. Changes in these courses do occur but such change usually fits under the current course description.

THE MANAGEMENT COMPUTER SYSTEMS PROGRAM: AN INTERDISCIPLINARY APPROACH

The Management Computer Systems (MCS) program at the University of Wisconsin-Whitewater is a true inter-college undergraduate CIS type major as defined in the recent DPMA Model Curriculum (1). Faculty from the Department of Mathematics and Computer Science in the College of Letters and Sciences have joined with their colleagues from the Management Department in the College of Business and Economics to form a degree program which produces graduates with a strong technical background oriented toward business systems applications in a commercial environment. Because of the dual college involvement, students may earn either a BBA or a BS degree in this major. The core computing courses required for each of the two degrees are identical; the only differences lie in the general requirements of the two individual colleges.

OVERVIEW OF CURRENT CURRICULUM

The MCS Program has evolved since its inception in 1978. The Program as it existed in 1981 has been previously published (2). The MCS Program is now the first recipient of the DPMA Educational Foundation International Four Year Institution award (9). For completeness a shortened listing of the current program is included.

Prerequisite courses are a typical business math course including topics such as interest, annuities, linear and quadratic modeling, matrices and linear programming; and an Introduction to Programming course using structured BASIC and FORTRAN and covering topics such as: while statement (loops), if-then-else statements, arrays - single and two dimensional, character manipulation, subroutines, files, pseudocode, and program documentation. Structured programming is used with an emphasis on maintainability and documentation.

CORE COURSES

The seven core courses have been designed so that a student can traverse them in four semesters, although students frequently require five or six semesters to incorporate extra computing and business electives into their program while maintaining a reasonable work load.

CONCEPTS AND PROGRAMMING OF MCS1. Similar to DPMA recommendation: CIS-2 (Application Program Development I). Similar to a portion of ACM recommendation: IS2 (Program Data and File Structures). This first course in the major proper uses PL/I as a vehicle to teach business programming concepts. Proper structure is extremely important. Application topics include report generation, array processing, data validation (edit), sequential files, and file merging. Background topics such as

corporate DP structure, career paths and the history of data processing are covered.

CONCEPTS AND PROGRAMMING OF MCS2. Similar to DPMA recommendation: CIS-3 (Applications Program Development II). Similar to a portion of ACM recommendation: IS2 (Program Data and File Structures). The student continues to learn programming concepts, data structures, and direct and indexed file structures. Assignments in this course are quite involved, traditionally the largest is a complex simulation using several stacks, queues, and linked lists.

CONCEPTS AND PROGRAMMING OF MCS3. Similar to DPMA recommendation: CIS-6 (Database Programming Development) and CIS-11 (Advanced Database Concepts). Similar to ACM recommendation: IS4 (Database Management). In this course COBOL is the vehicle used to access a commercial database. Our advisory board insists that we not use a "toy" educational database.

HARDWARE AND SOFTWARE SELECTION. Similar to portions of DPMA recommendation: CIS-8 (Software and Hardware Concepts) and CIS-14 (Information Systems). Similar to portions of ACM recommendation: IS1 (Computer Concepts and Hardware Systems) and IS8 (Systems Design Process). This course contains hardware architecture as well as basic principles for evaluating equipment and software. The student must use technical literature in making evaluations.

SYSTEMS ANALYSIS AND DESIGN I and SYSTEMS ANALYSIS AND DESIGN II. Similar to portions of DPMA recommendations: CIS-4 (Systems Analysis Methods) CIS-5 (Structured System Analysis and Design) and CIS-7 (Applied Software Development Project). Similar to portions of ACM recommendations: IS3 (Information Systems in Organizations), IS5 (Information Analysis), IS8 (Systems Design Process), and IS10 (Information Systems Projects). A two course sequence (3) that combines a thorough grounding in systems development with each student working on a team developing a real project (4), usually for a campus organization or office. Each team works closely with their user. Class presentations and discussions point out the different problems that occur with users of various levels of computer sophistication. Our advisory board feels that this two course sequence is the strongest aspect of the program. Projects are carried to completion including user training.

TOPICS IN COMPUTER APPLICATIONS. Similar to portions of DPMA recommendations: CIS-12 (Distributed Data Processing) and CIS-9 (Office Automation). Similar to portions of ACM recommendation: IS6 (Data Communication Systems and Networks). Rockart and Flannery, MIT, recently concluded (7), "There is little doubt in our minds that end-user computing will be the dominant segment of information systems in most large companies by the end of this decade." Wasserman, UCSF, and Gutz, DEC, agree, (8) stating, "...an increasing share of what we now term programming will be carried out by user-operators who will have tools at their disposal that permit them to interact naturally with a computer system." This course currently looks at packages as diverse as statistical packages, spreadsheets, graphics, microcomputers as stand alone and distributed workstations, and report generating packages all taught from an information center point of view (i.e., the students from this class will likely be resource personnel).

OTHER REQUIREMENTS

In addition to the core curriculum, students are required to select two courses from SIMULATION, OPERATIONS RESEARCH, BUSINESS DATA PROCESSING MANAGEMENT, ASSEMBLY PROGRAMMING, COMPUTER ORGANIZATION AND SYSTEMS PROGRAMMING, and MCS COOPERATIVE STUDIES. Under the MCS Cooperative Studies a student spends a seven month period in the information services area of an approved company. All MCS majors must take ACCOUNTING CONCEPTS, MANAGERIAL ACCOUNTING, MANAGEMENT CONCEPTS, and at least two additional upper level business courses. The BBA students must complete the common core of business courses required by AACSB.

The Business school also offers courses similar to DPMA recommendation: CIS-9 (Office Automation), CIS-10 (Decision Support Systems) and CIS-15 (Information Resource Management). There is not yet a course similar to CIS-13 (EDP Audit and Control).

BIBLIOGRAPHY

1. Adams, D. R. and T. H. Athey, DPMA Model Curriculum for Undergraduate Computer Information Systems Education. DPMA, 505 Busse Highway, Park Ridge, IL 60068.
2. Gerlach, J. and I. Goroff, The UW Whitewater Management Computer Systems Program. ACM SIGCSE Bulletin, Vol. 13, No. 1, pp. 171-176.
3. Goroff, I., A Systems Analysis and Design Course Sequence. ACM SIGCSE Bulletin, Vol. 14, No. 1, pp. 123-127.
4. Goroff, I., The Live Student Project. This publication.
5. Horton, R., An Industrial Survey of End User Computing Topics. This publication.
6. Nunamaker, J. F. Jr., J. Couger and G. B. Davis, (editors), Information Systems Curriculum Recommendations for the 80's: Undergraduate and Graduate Programs. Communications of the ACM, Vol. 25, No. 11, November 1982, pp. 781-805.
7. Rockart, J. F. and L. S. Flannery, The Management of End User Computing. Communications of the ACM, Vol. 26, No. 10, October 1983, pp. 776-784.
8. Wasserman, A. I. and S. Gutz, The Future of Programming. Communications of the ACM, Vol. 25, No. 3, March 1982, pp. 196-205.
9. Awarded at the Data Processing Management Association's International Computer Conference and Business Exposition, Anaheim, California, November 6, 1984.

GRADUATE EDUCATION FOR TRAINING INFORMATION SYSTEMS CONTENT

by

John F. Schrage
Southern Illinois University at Edwardsville
and
James Savage
General Dynamics Corporation

ABSTRACT

The noted graduate program in information systems has been training individuals in the field for over ten years with the given program updating the original program to provide specializations which are becoming an essential part of the information systems career ladder. The set of course and respective configurations were the work of the faculty and its industrial advisory committee. The common computer courses for the undergraduate core are intended in the beginning of the specialized degree. The three specialization tracks center on the traditional technical concepts with other tracks dealing with information systems for management and office automation. The research component of an advanced degree is also emphasized with a real design project. Additionally, a set of courses for the MBA student is specified to provide a minimum specialization for the management oriented graduate for the job market.

BACKGROUND

The original program was evaluated by the faculty and its industrial advisory committee with comparisons to their needs and other programs in information systems. An evaluation was completed on the current program and its relevance to the standard curriculum plans from the Association for Computing Machinery (ACM) and Data Processing Management Association (DPMA). That led to a comparison to what major universities were offering in a comparison with the current and standard curricula. Integrated in this was their personal feelings from hiring information systems professionals and what the advisory committee felt were the future trends in the information systems field. Independent from the committee, the faculty examined the literature on needs for the field and developed a curriculum plan. After about six months, the two groups presented their plans. Except for the constraint of hardware and software, the program from both groups, in respect to required topical content, were similar. The revisions in the curriculum lead to the following document and concepts.

The information systems program is a graduate professional degree program designed to combine business and computer systems skills within the private and public sectors. While theoretical concepts are taught, those topics are integrated into realistic business problems. Many of the topics from the program have been on the leading edge of the information systems field as noted in the practical application of topics such as information resource management, fourth generation languages, office automation, and structured analysis and design methodologies.

While students can enter the information systems program without business or computer coursework, they must meet the common body of knowledge (CBK) as described in the American Assembly of Collegiate Schools of Business (AACSB) in the process of taking the required coursework in information systems. The common body will be different in specific courses, but the content will align with the MBA program in respect to concepts. Coursework included in the pre-core dealing with information systems provides a common base for the information systems student. Some of the noted courses are provided to the MBA student as a specialization in the traditionally management oriented degree program.

THE MS in MIS PROGRAM

The graduate information systems program is divided into five segments -- (1) pre-core, (2) MIS core, (3) MIS specialization, (4) electives, and (5) design project. The pre-core, dealing with the common body in business and information systems, is waivable through the previously completed academic coursework from the undergraduate degree. The core consists of courses in advanced systems methodology, managing information resources, advanced file processing in a high level language programming or comparative analysis of programming languages, and principles of information systems theory. The specialization is from technical concepts, systems management concepts, office automation, or, in rare instances, a specialization determined by mutual written agreement between the student and the MIS graduate advisory committee. The electives may be any courses accepted for graduate credit. Only four elective hours may be in MIS, the other eight hours must be outside of the MIS designator. The culmination of the program will be the design sequence which will consist of a course in MIS research and policy and the design project paper.

PRE-CORE COURSES

The pre-core of business and information systems courses consists of nine courses with each waivable if specific content is taken from an accredited college or university with a B- or better grade in each course within seven years before acceptance into the program. The general business component of the pre-core consists of content as follows:

- (1) Accounting Fundamentals.
- (2) Survey of Marketing.
- (3) National Income Analysis and Policy.
- (4) Quantitative Methods -- either statistics or linear programming and calculus concepts.
- (5) Operations Management.

The computer and information systems component of the pre-core consist of the following four courses:

- (1) Computer Concepts. The overview of general computer concepts is given to provide a general literacy on applied computing.
- (2) Data Base and Communication Systems. Building on the prerequisite of the computer concepts

- material, basic concepts/terminology dealing with data base and communication systems are studied using the case approach to integrate the abstract to the real technical environment.
- (3) Structured Analysis and Design. With the minor introduction in the prerequisite material of computer concepts, the systems cycle is presented in both a traditional and structured approach with emphasis on structured methodologies.
 - (4) Programming Information Systems. Uses the prerequisite of the computer concepts material, for the computer programming in two languages, one being COBOL with concepts through files.

MIS CORE COURSES

The MIS core consists material in information systems theory and continuation of material in system design and programming. Using the basic computer concepts from the pre-core, the principles of information systems theory provides an overview of information systems theory within organizations including concepts for the MIS, DSS, and Expert systems. The advanced structured techniques examines advanced design concepts and functions plus financial control of projects and costs. In the programming area, the student can chose between a continuation of the pre-core programming or concepts dealing with a comparison of programming languages. The continuation programming course emphasizes the file (ISAM, relative, and virtual) and data base concepts using high level language with interactive programming. The comparative programming languages gives an analysis of major languages in the business environment with attention to use and environment.

THE MIS SPECIALIZATIONS

Each of the specializations consist of five courses to provide a concentration in technical concepts, systems management theory, or office automation. The specializations have a set of specific required courses plus a choice of selective courses and information systems electives.

Technical Concepts

The technical specialization consists of the following three courses plus two approved electives:

- (1) Data Base Management Systems.
The continuation of concepts from pre-core course in data base and communication systems and programming systems to emphasis the use of the data base design.
- (2) Communication Systems Design.
The continuation of concepts from the pre-core course in data base and communication systems with emphasis on communication systems design.
- (3) one of the following two courses:
 - a. Applied Operating Systems.
Using the theoretical concepts from the core programming course, the content with provide an application of theoretical technical systems concepts to the generic area of operating systems.
 - b. Fourth Generation Languages.
Using the systems theory of the principles course, the theoretical application of the "user programmer" with the macro languages available.

Systems Management Theory

The systems management theory specialization consists of the following four courses plus one approved elective:

- (1) Managing Information Resources.
Using information systems theory, the material will apply information as a resource within an organization for financial aspects plus research in MIS trends.

- (2) Decision Support Systems.
Using information systems theory, an in-depth treatment of the decision support process using appropriate tools for manager and computer professional.
- (3) Fourth Generation Languages.
Using the systems theory of the principles course, the theoretical application of the "user programmer" with the macro languages available.
- (4) one of the following two courses:
 - a. Information Systems Administration.
Based on the experience of systems analysis and design plus programming, the administration of the computer facilities from people to machines is examined in theory and practice.
 - b. Small Business Systems.
Using material from the basic concepts, the small business computers and their software use within the organization and the problems associated therein are studied.

Office Automation

The office automation specialization will consist of the following four courses plus one approved elective:

- (1) Office Automation.
Based on the theoretical base, the technological components of office automation are studied to include word processors, photo composition devices, micro form equipment, reprographics, facsimile devices, phone and mail systems and conferencing.
- (2) Office Information Systems.
Using the principles of information systems plus design, the systems approach for voice, text, and image applications in an office environment are utilized in the design of real-time environments.
- (3) two of the following four courses noted previously:
 - a. Managing Information Resources.
 - b. Decision Support Systems.
 - c. Fourth Generation Languages.
 - d. Information Systems Administration.

ELECTIVES AND DESIGN SEQUENCE

The electives in the program are meant to strength the student in a content area which will enhance the job opportunities upon graduation. Only one of the course can be taken in information systems to force the student to examine some minor set of courses outside of the "computer" realm.

The design sequence will consist of a course in research methodology which will set the stage for the final proposal and project. Various research methods will be examined to include graduate thesis requirements. The material will culminate with the design project for completion under direction of the MIS faculty advisor during the following term. The project will be the functional system design for an organization using information systems theory and design including specialization from the program coursework. The paper must meet the Graduate School thesis requirements and is orally defended.

MBA SPECIALIATION IN INFORMATION SYSTEMS

While the above explained the specialized degree in information systems, many students in the MBA programs are also looking for a set of course to enhance their career objectives with the computer as a tool. A set of course were developed from the information systems degree to provide specific content and giving an overview of the information systems field. The major reason for specific courses is that without specifics the MBA student takes some course and then tells employers he has the computer specialization. The MBA program specialization in

information systems has the following four courses as a minimum set of skills:

- (1) Data Base and Communication Systems.
- (2) Structured Analysis and Design.
- (3) Programming Information Systems.
- (4) Graduate Elective in information systems.

SUMMARY

While there is not any educational program which can will totally get the student prepared to walk directly into a position, the above program has been successful for our graduates. Business organizations have provided steady input into the educational process for information systems. Faculty and advisory board members have continued to be active in professional information systems groups to give a national rather than local view to the program. With some improvement in hardware to handle available software, the student will be able to have a more realistic view for that entry level systems position in not only the computer facility but information centers and user departments. Likewise, the MBA student is aware of the need for information systems skills as an enhancement to that career.

REFERENCES

- AACSB Accreditation Council. Policies, Procedures, and Standards.
- Athey, Thomas H. and David R. Adams, editors. DPMA Model Curriculum for Undergraduate Computer Information Systems Education. Park Ridge, IL: Data Processing Management Association, 1981.
- Benjamin, James; Wilbur Campbell; and John Schrage. "Career Oriented Graduate Program in Management Systems," SIGCSE Bulletin, 11:10-13, February 1979.
- Dickson, Gary W. and Ralph H. Sprague, Jr. A Short History of the Information Systems Area: An Academic Perspective. paper presentation at the First International Conference on Information Systems, Philadelphia, PA, December 1980.
- Couger, J. Daniel. "Curriculum Recommendations for Undergraduate Programs in Information Systems", Communication of the ACM, 16:727-749, December 1973.
- Nunamaker, Jay F., editor. "Educational Programs in Information Systems", Communications of the ACM, 24:124-133, March 1981.
- Nunamaker, Jay F., Jr., J. Daniel Couger, and Gordon B. Davis, editors. "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs," Communications of the ACM, 25:781-805, November 1982.

An Undergraduate CIS Program within a

Computer Science Department

Ali Behforooz Ph.D.
Onkar P. Sharma Ph.D.
Professors, Computer Science Dept.
Moorhead State University

Moorhead, MN 56560

Abstract: Traditional computer science programs offered by mathematics, engineering and computer science departments tend to become technical and science-oriented. At the same time, a large number of employers who hire graduates of these programs are less interested in the technical and engineering training that these graduates have acquired. Their interest lies heavily in applications of computers in the business environment. It was for this reason that DPMA came up with a model curriculum [1] for a new major called Computer Information System (CIS) and the ACM came with its own curriculum suggestions [2] a year later. In this paper we discuss the syllabus of a CIS program developed at the Moorhead State University (MSU) within the computer science department. The highlights of the paper are the curriculum itself, the fact that it is offered by a traditional computer science department and the popularity of the program among both employers and students.

Acknowledgement: This program was developed under a Bush Foundation grant. Other participants in the program included Drs Curtis Bring, Martin Holoien and Benjamin Lin. Significant contributions were made by Dr. Gerard Morris from the Computer Science Department and Drs Samuel Roy and Clyde Vollmers from Business Administration Department. Our thanks also go to Dr. David Newman from the University of Minnesota, Business Administration Department who served on this project as an outside consultant.

INTRODUCTION

Moorhead State University (MSU) is one of the seven in the Minnesota State University system with an enrollment of more than 6000 students. The university is situated in the Red River Valley on the Northwestern border of Minnesota with North Dakota. The university offers a diverse liberal arts curriculum leading to the bachelor's and master's degree in over 100 major fields. The computer science department, which is part of the division of social and natural sciences, was established in 1970 and offers students a choice of the following programs:

- a major in computer science, BS degree
- a major in computer science, BS degree, teaching
- a major in computer information systems (CIS), BS degree
- a minor in computer science, business-oriented
- a minor in computer science, science-oriented
- a minor in computer science, teaching
- a minor in computer information systems
- a major in computer science, MS degree

The enrollment in the undergraduate program currently stands at 340 majors. The master's degree program was started in the fall of 1981. The current enrollment stands at 35 students. There are ten faculty members in the department and this number is expected to increase to 12 next year.

We have produced over 440 graduates currently working in the industry. Through constant contact with the employers and the graduates we have realized that there are two drastically separate groups of employers with different requirements.

Our department is responding to the needs of one group very well and they come back every year to hire more of our graduates. This group includes employers such as IBM, Univac, CDC, Texas Instruments, Honeywell, AT&T, or SoftTech. When we talk to these employers, the comments we receive are very encouraging. They are very happy with our graduates and would like to hire more of them. The comments we hear from students working at these places are also always good. They are proud of the training they have received and think they are well placed. The only suggestion they have is that more courses in oral and written communications should be required.

On the other hand when we talk to the other group of employers which may be mostly categorized as small business and primarily data processing shops such as banks and insurance companies, we receive comments such as:

--we wish they knew some accounting; they cannot communicate with the head of our accounting department.

--we wish they had some background in management; they have hard time understanding the needs of the management.

--we wish they knew some basics of finance; inventory control, scheduling, etc.

--we wish they had better training in Systems Analysis and Design.

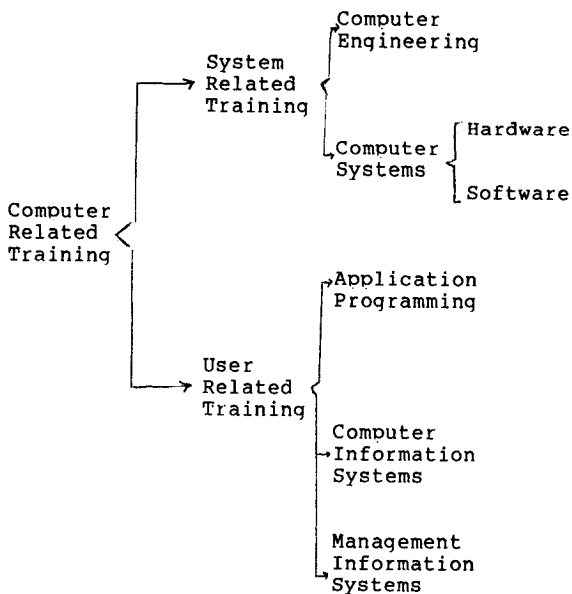
Graduates employed by these companies refer to some courses they have taken which are not helpful at their work. They also mention some topics that they think would have been more useful for them. In particular, they mention the same set of topics that the employers of the second group did.

Studying all of these comments from the employers and the graduates with reference to our computer science curriculum, we realized that the problem was not something that could be solved by adding a course or changing the graduation requirements. The problem deserved a more in-depth solution. It was at this point that we decided to train two separate groups of graduates for serving two separate groups of employers.

This led to our decision of starting a new program in Computer Information Systems to be designed primarily to serve the needs of the second group of employers. We mainly used the guidelines set by the Data Processing Management Association (DPMA) [1] and the Association for Computing Machinery (ACM) [2]. However, the program we developed is somewhat different from both of these curriculum recommendations but follows the major guidelines recommended by both.

COMPUTER-RELATED TRAINING AT UNDERGRADUATE LEVEL

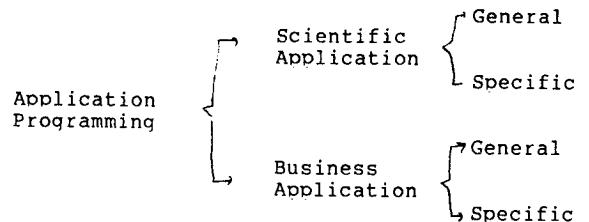
The following diagram is a general representation of computer-related training available at different schools in the nation.



Computer Engineering training is conducted primarily by the Engineering departments under the Computer Engineering program. The graduates of this program mostly work at manufacturing firms mainly producing computer-related hardwares and electronic circuits.

Systems training is conducted primarily by the computer science departments under the computer science program. The graduates of this program are well trained for systems development work. Most jobs within the computer systems manufacturing firms are held by these two groups of graduates.

The application programming jobs are typically held by persons with programming background. During the 60's and early 70's almost anyone who could write a program in COBOL, FORTRAN and/or RPG would be hired as an application programmer. These days, application programming has a completely different meaning. The following diagram is intended to show the separate branches under the application programming.



Specific scientific application programming refers to jobs that require programming for a specific application within an individual firm. The same is true for specific business application programming. The demand for specific business and/or scientific application programmers is decreasing since the late 70's. This is mainly because of the availability of the canned programs and sophisticated packages for almost any application. The general application programming refers to the programming efforts whose end results are canned programs prepared for the public. Whatever demand has been lost in specific application programming has been more than gained in the general area.

In the general and/or specific scientific application programming, the employers hire computer science and computer engineering majors or they hire science, mathematics, physics and chemistry majors with strong computer background. But the needs of the general or specific business application programming are not being served by these graduates. The employment is shifting from the science/engineering graduate to CIS or MIS graduates.

A Computer Information Systems (CIS) program is concerned primarily with the areas such as information systems design, database design and the application of systems development life cycle to computer-based systems. It should be the goal of a CIS program to train students such that they are able to communicate with all the departments within a company, understand their information-related needs and respond to them. A Management Information Systems program (MIS) is concerned primarily with the areas such as management of information systems, database management, and the study of the organizational behaviors with respect to information technology. A MIS graduate should be able to integrate the computing environment of a company with the organizational goals.

THE CIS PROGRAM AT MSU

Objectives:

The primary objectives of the CIS program at MSU include the following:

1. To give proper training to students who intend to work in business application programming environment.
2. To prepare better-qualified graduates meeting the current needs of local and national employers.

3. To upgrade the computer application aspect of the computer science department, by making available some of the CIS courses to the regular computer science students.
4. To provide opportunities for students in our Business College to acquire computer information-related background as it directly applies to their major study.

These objectives have been partly derived by examining the career opportunities for the CIS graduates. The following are the two most common jobs held by them.

1. Application programmer/analyst:

A person holding the position of a programmer/analyst should be able to:

- function as an individual or as a member of a group.
- work with the user of an Information System.
- function effectively in the planning and development phase of an Information System.
- fully understand the concept of the systems development life cycle.
- function as a project manager.
- function as a dynamic person when it comes to learning and changing.

2. Information Systems Specialist:

This includes areas such as information system planning, design, management and administration. The ACM recommendation lists a table of needed knowledge and abilities for working effectively in an information systems environment [2, Page 785].

Curriculum:

The CIS curriculum is divided into 3 parts: Part one is the requirements from related fields, part two is the requirements from the computer science and computer information fields and part three is the liberal art requirements. The liberal art requirements are decided by the liberal art committee of the university and usually an individual department doesn't have much to say about it.

Moorhead State University requires a total of 192 quarter credit hours for a four year degree. We specify a total of 119 credits for the CIS major. The remaining 73 credits satisfy the rest of the liberal art requirements and the free electives.

I. Requirements from Related Fields (63 quarter credit hours)

A. Accounting (12 credit hours):

We require a total of 12 quarter credit hours (3 separate courses) from accounting. The courses required are Principles of Accounting I and II, and Managerial Accounting. The main topics discussed in this sequence include the following: Fundamentals of Accounting, the recording process, preparation of financial statements, financial statement analysis, managerial accounting fundamentals, uses of managerial accounting data, and decision making.

B. Finance (4 credit hours):

We require a single course in corporate finance. The major topics covered in this course include the following: Analysis of methods used to manage assets, liabilities and investment, stockholders equity, inventory control and financial decision making.

C. Management (12 credit hours):

We require a full year course work in management. This amounts to a total of 3 courses for 12 quarter credit hours. These courses are: Principles of management, Organizational behavior and Management science. The major topics covered in these courses include the following: Fundamental topics in

management, planning, decision making, problem solving, productivity, organizing for stability and changes, leading and controlling, introduction to organizational behavior, individual processes, group processes and organizational processes, introduction to quantitative decision making and models, inventory models, linear programming and simplex method.

D. Communication Skills (15 credit hours):

In the area of communication and social behaviors we require the following courses:

--A course in Technical Report Writing (4):

The major topics discussed in the course are: Expository writing dealing with scientific subjects and planned for a specialized audience, documenting, writing abstracts and preparing reports of original investigation.

--A course in Effective Business Speaking (3):

The major topics discussed in this course are: Theory and practice of speeches prepared for business and professional audiences and practical aspect of speaking done by business and professional people.

--A course in General Psychology (4):

This is an introductory course in general psychology including a survey of content and methods of modern psychology.

--A course in Social Behavior (4):

The major topics discussed in this course are: The study of influence of other people on the behavior and attributes of individuals, attitude changes, the effect of being part of a group, aggression, attraction, sex roles and discrimination.

E. Economics (8 credit hours): We require two courses in principles of economics. The first course discusses the micro economics and the second one the macros. The major topics discussed in these courses are: An introductory study of microeconomics with emphasis on the price system, resource allocation and income distribution, an introductory study of macroeconomics with emphasis on national income, fiscal and monetary theory and policy, unemployment and inflation.

F. Mathematics (12 credit hours): We require a sequence of three mathematics courses beyond algebra and trigonometry. The major topics covered in the first of these three courses are: Functions, matrices, linear programming and the simplex method, limits, derivatives, maximum and minimum. The next two courses cover topics in applied statistics and those are: Descriptive statistics, probability, distributions, sampling, estimation of means, test of hypothesis, confident intervals, linear regression, multiple regression, correlation, analysis of variance, time series and nonparametric tests.

II. Computer Science and Information Systems (56 quarter credit hours):

A. Business Data Processing (4 credit hours):

An introduction to basic concepts of digital computers and their application to business with emphasis on microcomputers application.

B. Introduction to Computers and Programming (8 credit hours): Introduction to problem solving, algorithm development, algorithm representation, data representation and organization in computer memory, structured programming, programming in a high-level programming language, introduction to hardware and software concepts.

C. COBOL and RPG Programming (6 credit hours):

A discussion of COBOL and RPG programming language including file processing and report writing.

D. Computer Systems (4 credit hours): An overview of computer organization and systems software, hardware component of computers, the organization of computer and the systems' software such as compilers, assemblers and operating systems.

E. Structured Program Development and Data Models (4 credit hours): File processing, Structured program development and problem solving emphasized through the use of advanced features of COBOL, Overview of data models including hierarchical, network and relational models.

F. Structured Systems Analysis and Design (4 credit hours): Analysis and design of computer information systems including systems development life cycle and the the accompanying analysis and design tools.

G. Design and Implementation of Information Systems (4 credit hours): Introduction to the financial, technical, and strategic information systems planning processes emphasizing business goals, policies, plans and management style. Also includes means for selecting large system projects assessing the current state of installation and review of hardware/software/ service resources.

H. Systems Analysis and Design Field Project (4 credit hours): Design and analysis of a practical real life information systems project. This course is taught by a team of people from CIS, accounting and management. It is designed to provide a practical view of major topics discussed in information systems analysis, design and implementation.

I. Introduction to Software Engineering (4 credit hours): A discussion of project management, software system design, verification, validation, security, privacy and legal aspects.

J. Database Program Development (4 credit hours): Introduction to application program development in a database environment with emphasis on loading, modifying and querying a database using a high level language. Also includes a discussion of database information systems, administration, management and selection.

K. Data Communications (4 credit hours): Centralized, decentralized and distributed computer systems. Impact of distributed systems on businesses explored through case studies. Implication of computer communications.

L. Statistical Analysis of Data (2 credit hours): A study of computer-based statistical packages such as SAS, SPSS, BMDP and Minitab for data analysis with emphasis on one such package.

M. Computer Simulation Models (4 credit hours): An introduction to computer simulation languages and their application to the development of real word simulation models.

From the curriculum description given above, it should be evident that our intention is to give a fairly wide background in business-related fields as well as in the area of computers and information systems. In this regard, our curriculum follows the ACM recommendations.

The Technical Environment:

Students have access to CDC cyber 170 time-shared system, which is equipped with a variety of languages and software packages. They also have access to a UNIVAC 1100 series with a large number of business applications packages. In addition to the two time-shared systems, they have direct access to a VAX11/780 machine, university microcomputer laboratory having about 125 microcomputers and a rich library of micro software. In addition to these facilities, the students can acquire access to IBM or

Honeywell facilities through the North Dakota State University at Fargo and Concordia College at Moorhead. The department, since the approval of the CIS major in the fall of 1984, would want to buy software packages such as LOTUS 1-2-3, FRAMEWORK, SYMPHONY and NOMAD. We are in the process of creating a data communication lab in addition to our existing hardware lab.

CONCLUSION

The time has come to attend to the needs of business application programming so that our graduates can function properly in business environments. A decade ago the employers were forced to overlook the shortcoming of their employees because of the greater need for program coder and shortage of trained personnel. This situation is changing because a large number of application programs are now available as canned packages. This has reduced the demand for program coders. Instead, the demand is going up for well-trained system analysts, program analysts, application programmers, information systems managers, information systems specialists, etc. We believe that we have started a program in CIS which includes the broad guidelines contained in both the DPMA and the ACM CIS curriculum guidelines and meets the needs and requirements of application programming jobs as they relate to business environments.

REFERENCES

- [1] David R. Adams & Thomas H. Athey, Editors, DPMA Model Curriculum for Undergraduate Computer Information Systems Education, 1981.
- [2] Jay F. Nunmaker, Jr., J. Daniel Couger, Gordon B. Davis, Information Systems Curriculum Recommendations for the 80s: Undergraduate and graduate Program'ACM Communication, vol 25, no. 11, November 1982.

THE AGONY AND THE ECSTASY OF TEACHING DSS

Dr. Kuriakose Athappilly
Associate Professor
Western Michigan University
and

Dr. William Leigh
Assistant Professor
The University of Southern Mississippi

Abstract

A course in Decision Support Systems (DSS) is being added to the curriculum in many schools of business. The authors of this paper have been teaching DSS courses for the past several years, and in this paper they relate some of their experiences in this endeavor.

DSS: An Emerging Concept

Managerial decision making is at crossroads today. In the past, decision makers in general did not rely very much on the capabilities offered by science and technology. Although the information from quantitative experts and computer technologists became increasingly available, the managers could not make use of it in their decision making efforts because often it reached them too late and it was too difficult to understand. Consequently, most decisions in the managerial sector were made either by hunch or intuition influenced by past experience or political or individual considerations. The consequences of these decisions were sometimes catastrophic and often irrevocable. Consequently, a concern arose among management and executives to

make effective decisions by utilizing the advances of today's science and technology.

Numerous approaches were made to accomplish this task of reaching the capabilities of science and technology, specifically those of computers, to managerial decision making. Among them, Management Information Systems (MIS), Management Science (MS), Operation Research (OR), and Data Processing (DP), are undoubtedly the most successful scientific endeavors. None of these approaches, however, has fully capitalized the contributions of computers and quantitative methods to make decision making as effective as possible. This is where Decision Support Systems (DSS) finds its place. The Figure 1 shows the evolution of the impact of computers on business decisions through a series of systems developed in course of time.

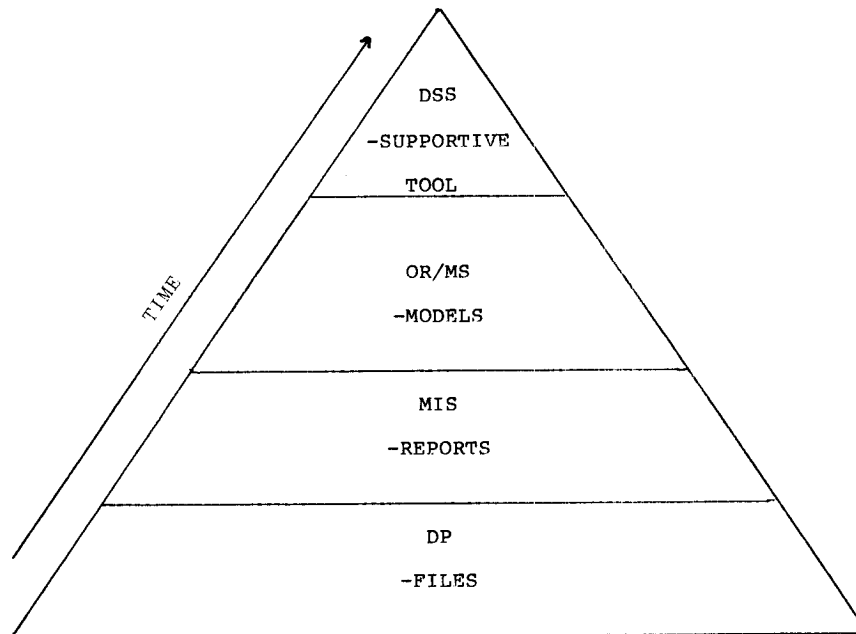


Figure 1. The evolution of the impact of computers on decision making process through DP, MIS, OR, MS, and DSS over time.

Broadly, DSS distinguishes from the above mentioned approaches in three distinct characteristics. These characteristics are (1) the impact of the system on decision making, (2) its payoff, and (3) the relevance to managers' decision making process. The impact of DP and MIS is mainly on structured tasks and their main payoff is the efficiency of the system by reducing time, cost, and energy. The relevance of these systems for managers' decision making is only indirect as they provide only some computer printouts in the form of reports.

The impact of OR and MS is mostly on structured problems. But structured problems place a heavy duty on decision makers by restricting them to the prespecified objectives, data, and constraints. Their payoff is to generate better solutions for the prespecified problems only. The managers found these systems helpful only in so far as these systems could provide detailed recommendations and methodologies for handling complex problems.

The impact of DSS, however, is on decisions themselves. The system is such that the computers and the analytic tools are at managers' beck and call for direct access and immediate results. The

payoff is effectiveness rather than mere efficiency. The relevance comes from the fact that the system is a supportive tool in the strict sense of the term. In other words, it does not replace human judgment. Because of these characteristic features, DSS becomes an attractive concept among managers and most influential and challenging among the researchers, practitioners, and educators.

An effective DSS should provide information support for managers at all levels in an organization to make decisions independently as well as interdependently through appropriate integration and coordination mechanisms. Consequently, unlike MIS, OR/MS, and DP, Decision Support Systems effectively support the top management strategic decisions, middle management tactical decisions, and lower management operational decisions. The Figure 2 explains the pyramidal structure of the role of management in structured, semi-structured, and unstructured decision environment. The uniqueness of DSS is that it is capable of assisting the decision making process, especially at the higher levels where there is less structure and less control of the decision variables in action.

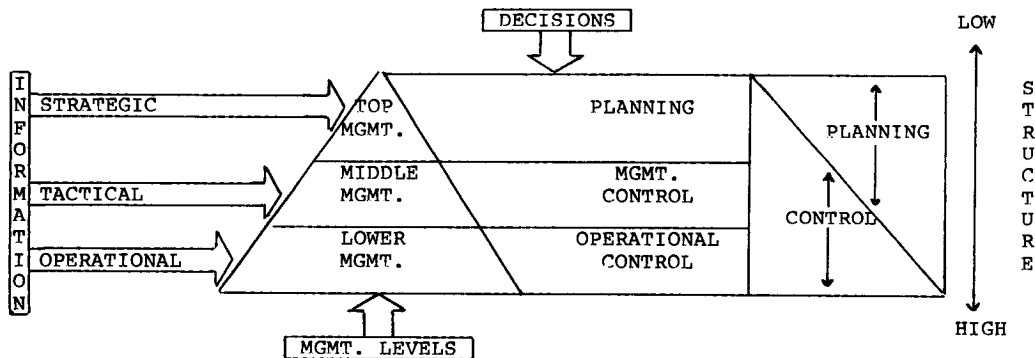


Figure 2. Management (MGMT) information/decision framework under planning and control in a DSS environment.

Many business enterprises have already implemented decision support systems and have experienced its advantages. The key success factor of DSS is the involvement of the decision maker in the design, development, implementation, and evaluation of the DSS system. Most of the managers today are not very much exposed to computers or computer based analytic aids. They are, therefore, often apprehensive or uncomfortable with the idea of DSS as it is a "computer-thing." Some of them, however, make a bold venture of facing the reality of today's computer technology and try their best to cope with its demands and challenges.

DSS in College of Business Curriculum

What has been discussed above being the scenario of the business world today, the educators in the field of Business Information Systems, have a vital role to play. This role tantamounts to the formation of a new U.S. corporate with a new line of executives and a new style of decision making. In order to make this vision a reality we, the Computer Information professors, need to educate our youngsters, the business students, with the theory and practice of Decision Support Systems.

Recognizing this need for exposing the concepts and applications of DSS, many schools of business introduced a course in DSS at the undergraduate and graduate level.

DSS for Non-CIS Majors

The second author of this paper has been teaching a DSS course as a core course in the business curriculum. This course is taken by all business majors, usually in their junior year. The thinking was that the DSS course is a desirable addition to the core course set because of its computer content and integrative nature. In general, this reasoning has proven to be valid.

Business students at the junior level have little preparation in decision theory, other management science, computer information systems, or the role of decision-making and decision-makers in business. Accordingly, a DSS course to this audience must lay all of its own foundations. Also, business students at this level are generally afflicted with a "multiple-choice" mentality, which makes it difficult for them to grasp concepts such as "semi-structured" problems. In common with the

other core courses taken in the junior year, such a DSS course must expose the students to problems which are not necessarily solvable by such closed form methods as they have learned to expect from their lower division textbooks in accounting and so forth.

It has been useful to structure the course around the decision-tree approach to problem-solving and decision-making: define the problem; generate alternative solutions; evaluate the alternatives and pick the best. The very idea that problems may need be defined by him is new to many students. The creativity necessary to generate alternative solutions is taxing. And the lack of a clear-cut method in many cases for the evaluation of alternatives is frustrating.

Use of computer aids must be an integral part of a DSS course. Problems must be defined and alternative solutions generated within the brain of the decision-maker, but a computer database can be used to supply an empirical basis for evaluation. A relational DBMS is learned and used by the undergraduates fairly easily. They use prepared databases for solving instructor-posed problems, and they design databases to be the centers of their own DSS's in their projects.

In working with the DBMS and with databases, the students become familiar with normalization of database schemas and with formulating relational algebra queries. The DSS context motivates this skill acquisition as it is a necessary part of the required decision-making exercises. Implicit in this process is the students' realization of the problems that a DBMS solves.

To evaluate alternatives from historical data, the students must develop estimates of parameters and estimates of confidence. This necessitates some experience with statistical estimation and sampling. The statistical theory has been left implicit. It has been found useful to approach these estimation problems with resistant methods of exploratory data analysis. Such tools are intuitive (and the students have not developed any phobias concerning them such as they may bring with them from their statistics course concerning confirmatory statistics). These resistant statistical tools are available in the package MINITAB. The students are supplied with an interface between their relational DBMS and MINITAB so that they can analyze their historical data with computer aids.

A usually successful classroom example of this type of databased decision-making is horse-race-betting. The Daily Racing Form is a database. The problem may be defined variously: "pick the winner"; "do not lose money"; maximize expected gain"; and so forth. The alternatives are the horses available for betting. At this level, the possibility of betting on several horses or the various types of bets are not investigated. This example can be naturally expanded into a field trip to do real decision-making (at the track).

The business context makes decision-making easy in one respect: the objective is always to maximize profit. A discussion of the ethics of the assumption can make for a few lively minutes, but nowadays the appropriateness of this assumption is rarely questioned.

An example of multi-criteria decision-making which the students relate to is the choosing of a personal car. Generation gaps open as the factor weights for economy and sportiness are developed in a group

process. In a young group, the Datsun 280ZX usually dominates the other possibilities.

Lecture on the above subjects is complemented with computer exercises. The mainstay of the course, however, has been the project. The project assignment has evolved into a tasking of the student to re-make a well-known decision from the past. A favorite is Hitler's invasion of Russia. For this particular decision, Hitler's problem (as it is thought that he saw it) must be defined, his alternatives outlined, and an evaluation of them done. This is to be done using historical sources for data. Finally, the students are to design a DSS (using their DBMS and MINITAB) and the necessary database which Hitler should have had.

Early on the students were allowed to find their own decision to make for the next project. This was a mistake as many students found that CONSUMER REPORTS was using exactly the same decision-making methodology.

DSS for CIS Majors

In the DPMA Model Curriculum, DSS (CIS-10) is classified as an elective. However, the prerequisite for the course is listed as Introduction to Information Systems (CIS-1). Accordingly, it seems that this DSS course for majors could not differ much from the course for non-majors described above.

Even if the prerequisites are strengthened to include programming or even systems analysis/design, the students would still lack an appreciation of decision-making contexts or management-science/decision-theory, which are really more central to the DSS idea.

The value of CIS-10 to CIS majors could be enhanced by the presence of other business majors in the class. This would enable group projects where the group members could participate in a role consistent with their chosen major. This type of experience would be beneficial to all, but especially to the CIS major whose other CIS coursework involves working in groups to only a major extent and then only in technical groups.

Conclusion

Thus, teaching DSS at undergraduate level is quite an agony in the real sense as it deals with many theoretical concepts, technical skills, and real world applications with which the students are not familiar. The course is carefully designed to be a challenge. Surprisingly, in spite of the rigor and heavy demands, the course is growing in acceptance and popularity. There are many reasons for this increasing popularity. First, is the promise that DSS offers for future employment. DSS is the new wave in information systems, and many of the leading business organizations have introduced DSS as the most important support system for decision making. Most of the interviews for jobs bear witness to this fact. The students sense this climate and are encouraged by it.

Second, is the opportunity that DSS offers the CIS students for a career that is more challenging and exciting than that of a programmer/analyst. DSS is executive mind support. It enables the decision maker to utilize computers at their desk top. While their fingers move on the keyboard, they can retrieve data, simulate situations and find different alternatives using "what if" questions interactively. Thus, the new line of students with

the necessary knowledge and skills in DSS will be able to assume managerial positions and utilize the advances made by today's science and technology to the fullest potential for effective decision-making. That is what the students want to hear about and want to practice in the future. Many students, after the completion of the course, decide to make their career in the emerging field of DSS.

Third, is the converging character of DSS which brings many interrelated disciplines into a meaningful focus. The students often say that this course brings together all of what they have learned during their entire student life. As we might already know, to be a successful DSS expert, one should have an appreciation and understanding of computers and computer-related quantitative models and techniques on one side--the "techy" side--and of the managerial decision making process which intertwines with human engineering, behavioral studies, psychology, communications, and logic on the other side--the "softv" side. A vast majority of CIS major students think that this course has to be the capstone course in the college of business curriculum.

Fourth, is the emphasis of the course on practical applications. The students have to go out into the real world and actually design, develop, implement, and possibly evaluate a DSS system in close interaction with the concerned public or private enterprise. This is a real challenge! The

students, however, love to have this real world experience. By doing these projects they get a realistic foretaste of the kind of things they have to do in the future and experience a great sense of accomplishment. Above all, they develop a sense of confidence which, otherwise, is very difficult to cultivate.

Fifth, is the missionary appeal of DSS. DSS is still a new concept to many. The students know that this concept will be attractive to managers as it can be a corporate's life-line to success. They know that if they put this idea properly they can sell themselves as future managers much more effectively than by any other means or methods presently available in the field of computer information systems. However, since DSS is still in its infancy, just like the apostles of the New Testament, the students have to go and preach the "good news" by "spelling relief as DSS to the frustrated management." By doing so, they become the champions of a great cause of bringing about a new U.S. Corporate, with a new line of executives and a new style of decision making. This missionary character is, perhaps, the most important factor for the popularity of DSS course.

This enthusiasm and zeal of the students is what brings in all the excitement and ecstasy of teaching DSS.

TEACHING CIS-15, INFORMATION RESOURCE MANAGEMENT

SANDRA E. POINDEXTER

NORTHERN MICHIGAN UNIVERSITY

This paper is intended to serve as a practical guide for individuals planning to teach Information Resource Management, CIS-15 in the DPMA Model Curriculum. Except for the course outline listed in DPMA Model Curriculum for Undergraduate Computer Information Systems Education very little material is available and no university presently offering the course was located. Topics covered are: Idea Generation, Gathering of Resources, Selected Approach, and Future Plans for CIS-15.

GATHERING OF RESOURCES

IDEA GENERATION

Northern Michigan University (NMU) offers a four-year degree in Computer Information Systems (CIS) originally developed by modifying other institutions' programs to fit our regional needs. Over time the curriculum has been revised and presently follows the Data Processing Management Association (DPMA) Model Curriculum to a point. The seven CIS core courses and eight business support courses are contained within the NMU curriculum. The three remaining computer courses are chosen from an advanced course of an Applied Software Development Project, a Decision Support Systems course, and selected computer science classes taught through the Mathematics and Computer Science Department.

It was the opinion of both CIS faculty and students that a variety of more advanced topics needed to be offered to keep the program current. The fact that the Mathematics and Computer Science Department was moving ahead in the area of advanced computer science topics made the issue even more apparent. The reason for the delay was the constraint felt by many educational institutions: faculty staffing. Teaching of the core courses in the program required all our available full and part time faculty.

The standard procedure at NMU for instituting a new course is to offer it as an experimental Special Topics class on a one-time basis. The faculty member supporting the course must obtain departmental approval by establishing need and relevance. If the course is successful, a series of approvals is required for it to be given a permanent status.

Information Resource Management, CIS-15, was my choice among the elective topics listed in the DPMA curriculum for a Special Topics course at NMU. This decision was based upon my personal preferences, an analysis of student needs, and the philosophy of my department, "Management, Marketing, and Computer Information Systems." With staffing constraints, I felt a course related to management would have a better chance of receiving departmental approval.

The format for a Special Topics class is quite flexible. I arranged the class as a two credit, seven week (1/2 semester) course. The class size was set at a maximum of thirty students to provide an environment for discussion rather than lecture. A two-hour time block was chosen for each class period to allow enough time for films to be shown and thoroughly discussed. As of this writing the course is scheduled to be taught January - March, 1985.

While I realized the DPMA model curriculum was still quite young, it did not occur to me that I would have difficulty obtaining materials for any course listed in it. I felt that with only a few phone calls I would be able to locate several faculty members around the nation who had taught the class and would gladly share their resources with a colleague. My first telephone contacts were with EDSIG (DPMA's Special Interest Group for Education) officials and educational regents of DPMA. While everyone I spoke with was very supportive and assisted me as much as they were able, I found no one who had ever taught the course. It was then necessary for me to develop it without much assistance.

I found that DPMA Model Curriculum for Undergraduate Computer Information Systems Education is an invaluable tool for topic selection. It supplies course descriptions, content, and selected references for each of its CIS courses. The NMU library holdings of periodicals such as Datamation, Infosystems, and Data Management, and special topic books are abundant. However, comprehensive textbooks and visual aids are scant. There are several MIS textbooks on the market, but I located only two books which dealt with the management of computer resources. I chose Information Resource Management, written by Hussain & Hussain and published by Irwin, Inc. in 1984. Its preface references CIS-15 as a text objective. The second text, also published by Irwin, Inc., is Corporate Information Systems Management: Text and Cases.

Case studies are sold individually through HBS Case Services, Harvard Business School, Boston, MA 02163 (previously called Intercollegiate Case Clearing House). A catalog of available cases and their descriptions should be found in the reference section of most libraries, or can be ordered from HBS Case Services. Since the cases can be purchased individually, it is an inexpensive way to obtain two or three comprehensive case studies. The length of the studies averaged approximately 20 pages. Books of MIS case studies are available through various textbook publishers and may be less expensive for students when the instructor plans to use many cases. However, not all MIS cases are appropriate for CIS-15. I chose two HBS case studies recommended in DPMA Model Curriculum: "First American Bank (B)," #9-176-001, and "Quality Life of New York," #9-179-200.

Finding suitable visual aids was the most

difficult task. Media Review Digest is located in the reference section of the NMU library. It lists films, video cassettes, and filmstrips by topical order which are available for purchase or rental. Most of the computer related material listed was at the introductory level. I also reviewed the catalogs of distributors of professional training materials such as ASI and DELTAK. While there were some appropriate films, the rental prices, which includes all workbook aids, were too high for our department's budget. My next resource was DPMA International which has a series of video cassette programs and slides available through their regional officers. Since NMU has a DPMA student chapter, the films are available free of charge on a loan basis. However, in NMU's region the films are so heavily in demand that I was unable to obtain them for a specific date and for most of the films a ten month notice was needed. I explained the dilemma to an individual at the DPMA International Headquarters and consequently have purchased several films directly from them. These purchases guarantee film availability when needed. The five films I plan to use are: "Computer Abuse - Computer Security," "Data Processing - The Manager's Role," "EDP Quality Assurance," "Motivating the Data Processing Professional" all from DPMA, and "Cloak and Data" from Vision Film Assoc., 85 Scollard St., Toronto, Ontario, Canada. I was able to obtain a copy of this last film on loan from the accounting firm of Ernst & Whinney.

SELECTED APPROACH

My plan for this experimental course is to provide a forum for discussing issues specific to the management of a computer information systems department. I feel a one-way presentation for this type of course, as in a straight lecture format, would result in little thought stimulation among students.

CIS-15, as described in DPMA Model Curriculum, is intended to be a full semester's course. Obviously, with only half the time some topics had to be eliminated. Figure 1 is a Gantt Chart for the topics selected. It is tailored to the chosen text, but covers what I consider to be the critical subjects in the CIS-15 course content. Interspersed throughout the seven week period will be the five films and three guest speakers. Through the use of the guest speakers, I hope to provide students with the perspectives of a Director of Information Systems, a computer vendor representative, and an attorney.

The prerequisites for CIS-15, should ideally be the CIS core and most of the business support courses, as upper level students have a better knowledge foundation and tend to be more participative in class discussions. However, the majority of courses taken by the CIS major at NMU are four credits, and I was unsure of student response for a non-four credit course. Seniors may not be able to fit it into their almost-completed degree. I also did not want to entirely exclude computer science majors. Therefore, I set the prerequisites as two CIS or CS courses and Principles of Management. Since the course is being offered as a Special Topics course I am free to alter prerequisites if it should be offered again.

The students' requirements are to be text readings, class participation, two case study analyses, and two short research papers. The case study analyses will be based upon the two outside Harvard Business School cases previously mentioned. The research papers will be four to

five pages and written on topics the students select from the course plan as indicated on the Gantt Chart. I will expect a minimum of two resources to be used in writing each paper. The Gantt Chart indicates when each of the written assignments will be due.

Oral presentations or walkthroughs have always been a mandatory part of my courses. However, with only seven weeks there was no time available for this important aspect. In a four credit version of the course I would definitely include a student presentation requirement.

No examinations are being given in this seven week course, again due to time constraints. However, the syllabus clearly states that if class preparation and participation are determined to be unsatisfactory, an essay exam will be given.

FUTURE PLANS FOR CIS-15

Once Information Resource Management has been offered as a Special Topic, the student course evaluations and my perceptions of its success will be discussed with the other CIS faculty. If we feel it is a course which should be expanded to four credits and offered on a regular basis, we will jointly submit a proposal to our department requesting such action be taken. No doubt some revisions and refinements will be necessary. An adjustment in topics, requirements, format, or prerequisites are all possibilities with an experimental class.

It is the goal of the NMU CIS faculty to offer one new CIS upper level course each academic year. Using this approach we will not unduly tax any of the faculty members and within several years will have met the standards of the DPMA model curriculum.

FIGURE 1
 INFORMATION RESOURCE MANAGEMENT
 SEVEN WEEK COURSE OUTLINE

CLASS PERIODS (2 HOUR BLOCKS)

TOPICS	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Introduction Film "Data Processing- The Manager's Role"														
Project Planning														
System Development														
Project Management			1											
Organizational Structure														
Distributed Processing														
Computer Center Mgmt Film "Motivating The Data Processing Professional"						2								
Computing Services														
Quality Control Film "EDP Quality Assurance"														
Auditing									3					
Standards														
Budgets														
Privacy/Security Film "Computer Abuse, Computer Security" Film "Cloak & Data"												4		
Computer Law														

Notes:

1. First research paper due on selection from topics 1 - 7 on above outline
2. Case analysis due on "Quality Life of New York"
3. Second research paper due on selection from topics 8 - 13 on above outline
4. Case analysis due on "First American Bank (B)"

TEACHING STRUCTURED COBOL PROGRAMMING USING A MODEL

RON TEEMLEY

DeVRY INSTITUTE OF TECHNOLOGY

This paper presents a method of teaching students to design and write structured COBOL programs using a model. The model is a detailed module chart that includes a module for each task required in a typical program. The MODEL method essentially eliminates the need for flowcharting and pseudocode. Several program designs based on this method are included.

Prologue

This paper presents a method of teaching structured COBOL programming using a "standard" or "model" module chart as a guide. The method could surely be used with any other structured programming language as well.

The motivation for this project is based on the fact that beginning programming students do not like to write pseudocode or flowcharts before they start coding a program. There is a valid reason for this: the first programs a student writes are so simple that there is no need for these guides. The students then get the mistaken idea that these devices are of little importance. What most students do is draw their flowcharts or write their pseudocode using their code as a guide! And the only reason they do this design work after the code is written is because the teacher counts this documentation as part of their grade.

Many students that follow this pattern get into trouble about their third programming course, when the programming assignments get more difficult. Then they are helpless because they never learned how to design a program when they had easy programs to work with.

Hence the search began for a method of program design that would be appealing and relatively painless to students as well as even fun at times! The resulting design techniques are based on a so called MODEL module chart: a single, detailed program design that is general enough to handle a large percentage of the programs a student will write.

This method is being used to teach COBOL I and COBOL II at DeVRY Institute of Technology in Dallas. The method seems to be a real "natural".

The Module Chart: A Natural Way to Represent both Procedure and Organization (Refer to Table 1)

The students first learn that the module chart is a very natural way to represent any procedure or organization. This is usually called a "top-down" or hierarchical presentation. The most well known use of these diagrams is to show a company or office organization (see Figure 1, Doctors Office Organization). But any procedure can also be shown very clearly and logically using a module chart. This is demonstrated in the following examples:

Fixing a Flat Tire (Figure 1), Students Weekday Routine (Figure 3) and Writing a Term Paper (Figure 4).

The same "top-down" design is used whether the structure described is an organization or a procedure. The top or A level modules name the overall procedure or organization. The second or B level is used to divide the major item into its logical components or subordinates. Since the terminology is different for procedures and organizations the two have been portrayed in the following table:

THE TOP DOWN METHOD FOR CONSTRUCTING A MODULE CHART

<u>ORGANIZATION</u>	<u>PROCEDURE</u>
<u>A Level</u> List the head of the organization in this single module.	List the overall task to be accomplished.
<u>B Level</u> List all managers that report directly to the head.	Break the overall task into its several logical pieces.
<u>C Level</u> List all supervisors or personnel that report to each person on the B level.	Break each piece on the B level into its logical as required.
<u>D Level</u> Continue this procedure on the <u>D and lower levels</u> until at some point the personnel and tasks arrive at their natural lowest level.	

Module charts are very effectively used in disciplines other than computer science, for example see Zen and the Art of Motorcycle Maintenance pp. 86, 223 (1).

The "MODEL" Module Chart

Once the students are familiar with the basic usefulness and simplicity of the module chart it is easy for them to understand that a module chart is a "natural" device to use to portray the structure of a computer program. This is particularly true if the students have already had an introduction to programming using Pascal or BASIC. In fact every computer program whose purpose is to produce a report has the same basic design (see Figure 5). This is the MODEL design for a computer program. In the top or A level module is described the task of the entire program. In the B level modules we split this big task into its several logical parts. The C level is used to break each B level task into still smaller subtasks as required. The process

continues on the D, E and succeeding levels until a natural stopping point is reached. At some point it is obvious that to break the task down any more would be ridiculous!

There are some basic and very important rules for constructing these modules:

1. Each module has a descriptive, well thought out name that clearly explains that module's job.
2. Each module that lies along the underside of the chart (that is, it has no submodules) has only one task. Furthermore, this one task should be brief. The rule: if the task is too long or complicated to keep easily in mind without strain divide it into smaller modules on the next lower level.

This is the **KEY** to this method of design: any computer program becomes manageable when broken down into small enough pieces.

Here are a few examples of computer programs that have the same design as the MODEL: Create a List of Customers, (Figure 6); Create List of Daily Sales, (Figure 7); and Create Employee Gross Pay List, (Figure 8).

It is easy to see that the design given in the MODEL will work for any report generating program. A few changes will obviously be required from one program to the next but the point is the basic design remains the same.

Teaching the COBOL Language one Module at a Time

The MODEL is also a very natural device to use to present the COBOL language once the data definitions and introductory parts of the language have been thoroughly discussed!

The idea is this:

1. Choose a well known programming situation. Create Employee Gross Pay List, Figure 8, is the example used here.
2. Introduce the COBOL commands typically used in each module as they are encountered. Discuss the modules in the top down order they appear on the module chart. For example the code for the A level module in Figure 8 would be:

A-CREATE-EMPLOYEE-GROSS-PAY-LIST.

```
PERFORM B10-START-UP.  
PERFORM B20-WRITE-HEADINGS.  
PERFORM B30-READ-PAYROLL-CARD.  
PERFORM B40-GROSS-PAY-PROCESSING-LOOP UNTIL END-OF-FILE-  
SWITCH = 'ON'.  
PERFORM B50-FORMAT-WRITE-TOTAL-LINE.  
PERFORM B60-CLOSE-DOWN.  
STOP RUN.
```

The first COBOL commands the students are exposed to are PERFORM and STOP RUN. This is not the usual way of going about it! But the fact is this is the most natural approach. In structured COBOL the PERFORM statement is the one that controls the entire program and ties everything together. The PERFORM statement ensures each module gets executed the correct number of times, in the correct order. The approach also enforces the importance of the A level module with respect to the B and lower level modules.

The COBOL language unfolds naturally as we discuss the other modules:

```
B10 - START-UP.  
OPEN INPUT PAYROLL-CARD-FILE  
OUTPUT GROSS-PAY-LIST-FILE.  
MOVE CURRENT-DATE TO DATE-HEADING.
```

```
B20-READ-PAYROLL-CARD.  
READ PAYROLL-CARD-FILE INTO PAYROLL-CARD AT END MOVE 'ON'  
TO END-OF-FILE-SWITCH.
```

```
B30-WRITE-HEADINGS.  
WRITE THE-PRINT-LINE FROM HEADING-LINE-1.  
AFTER ADVANCING PAGE.  
WRITE THE-PRINT-LINE FROM HEADING-LINE-2  
AFTER ADVANCING 2 LINES.  
MOVE SPACES TO THE-PRINT-LINE.  
WRITE THE-PRINT-LINE.  
MOVE ZEROS TO LINE-COUNT.
```

NOTE: This sample program used to teach COBOL can be rather "complete" in that it includes writing headings, doing arithmetic, etc. or several samples of increasing complexity can be used. The first sample could simply introduce control and input/output operations. Later programs could introduce the other instructions a "module at a time" by adding writing headings to the second example, then doing calculations in the third sample, etc.

Other Program Designs Using the MODEL as a Guide

Many other types of programs that would appear to require a different design are actually just variations of the MODEL: 1) MODEL + Additional Structure or 2) MODEL with single tasks replaced by a group of tasks.

For example, to construct Figure 9, Validate and Create Patient Disk File, start with the MODEL, delete module B50 and change C10 from a calculations task to a validation task.

Figure 10, Print Student Report Cards, is a single level control break program. It is the design of the MODEL with two additions: B40, a module to reset the control field and a group of modules to handle the control break (C10 and subordinates) and one change: writing the totals is replaced by a compute and write grade point average module.

NOTE: This design for a control break program is a true natural. **No** changes in the design are required if we go from 1 level of control break to 2 levels or two hundred levels!! Just reset all control fields at B40 and instead of having one group of control break processing at C10, line up as many groups of control breaks as is necessary on the C level: C10 (Minor control break), C20 (next lowest control break), etc.

Figure 11, Process Orders, at first glance looks very complicated but it is simply the design of the MODEL with Load and Unload table routines added at B30 and B80!

In Figure 12, Create Employee Profiles, the design of the MODEL is again retained; the final totals module is deleted from level B; the Processing Loop (B30) is quite extensive in this program so a new row has been added under B30 to first divide the Processing Loop into three parts; C10 Personal Processing, C20 Salary Processing and C30 Job Processing, and each of these parts has its own Process and Write steps!

Figure 13, Sequential File Maintenance, utilizes the Balance Line Algorithm presented by Cooper et al (2). Although this design was derived independently of the MODEL it is easy to see that it can be developed from the MODEL: on the B level add a module to choose the active key and delete the module to print totals; on the C level calculations are replaced by file maintenance tasks: build new masters and apply transactions to existing masters; finally the read master and read transaction modules are moved to be with the "build masters" and "apply transactions to masters" groupings respectively.

Random File Maintenance, Figure 14, is very close to the design of the MODEL: the calculations module is replaced by a set of eight modules whose purpose is to update the master file!

Epilogue

The MODEL approach to structured program design eliminates the need for most pseudocode and flowcharts. First, when the student sits down to design a new program the MODEL provides a starting point; the hardest thing about writing a new program is usually getting started! Second, once the student sees examples of other program designs derived from the MODEL he begins designing programs this way himself; adding, deleting and revising modules of the MODEL. Third, the code in each module is kept short so there is hardly ever any need to draw a flowchart or write pseudocode to help figure out the logic.

If flowcharts or pseudocode is needed to help design a troublesome module the students are encouraged to use them.

The MODEL method of program design is easy, fun, and it **works!!**

Table 1. LIST OF MODULE CHARTS

MODULE CHART NUMBER	NAME OF MODULE CHART
1	DOCTORS OFFICE ORGANIZATION
2	FIX A FLAT TIRE
3	STUDENTS WEEKDAY ROUTINE
4	WRITE A TERM PAPER
5	MODEL MODULE CHART
6	CREATE A LIST OF CUSTOMERS
7	CREATE A LIST OF DAILY SALES
8	CREATE EMPLOYEE GROSS PAY LIST
9	CREATE PATIENT FILE
10	PRINT STUDENT REPORT CARDS
11	PROCESS ORDERS
12	CREATE EMPLOYEE PROFILE
13	SEQUENTIAL FILE MAINTENANCE
14	RANDOM FILE MAINTENANCE

BIBLIOGRAPHY

- (1) Persig, Robert M., Zen and the Art of Motorcycle Maintenance, Bantam Books, N.Y., 1974, pp. 86, 223.
- (2) Cooper, R. H., D. D. Cowan, P. H. Dirksen, J. W. Graham, "File Processing with COBOL/WATBOL, WATFAC, 1973 (Waterloo, Ontario).

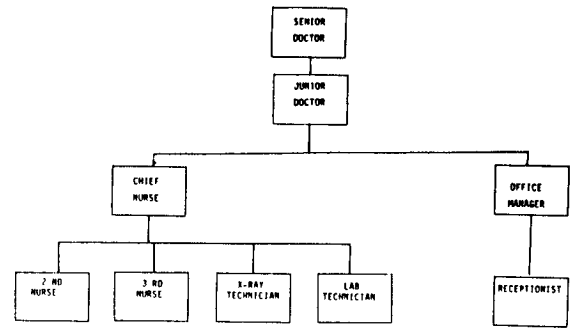


Figure 1 Doctor's Office Organization

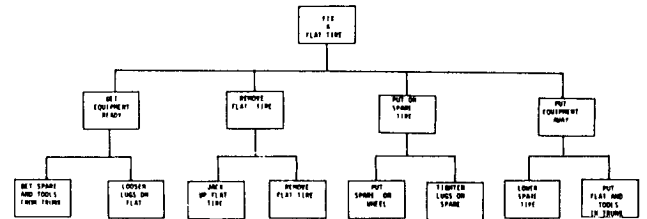


Figure 2 Fix a Flat Tire

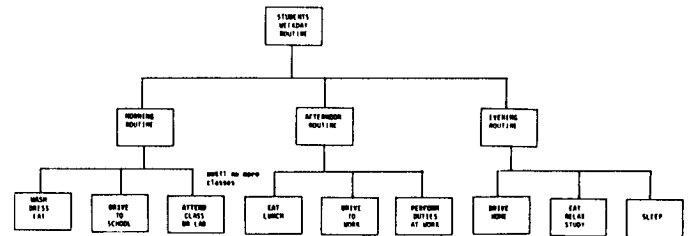


Figure 3 Students' Weekday Routine

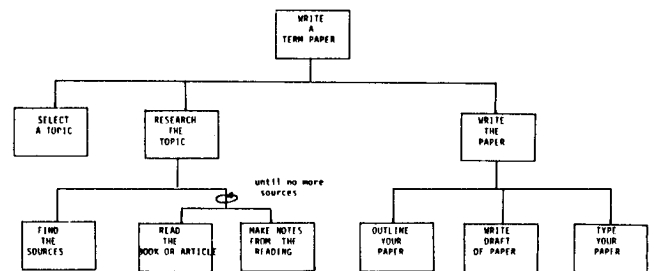


Figure 4 Write a Term Paper

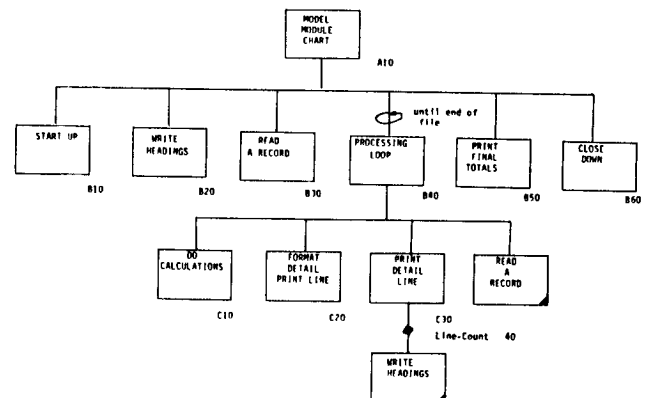


Figure 5 MODEL Module Chart

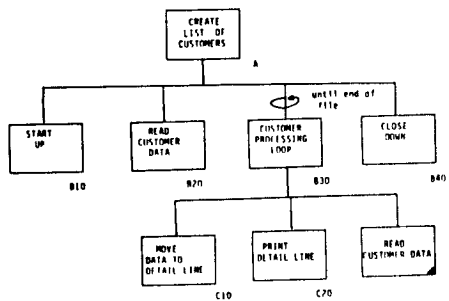


Figure 6 Create A List of Customers

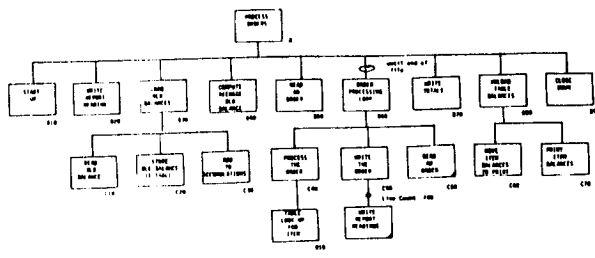


Figure 11 Process Orders

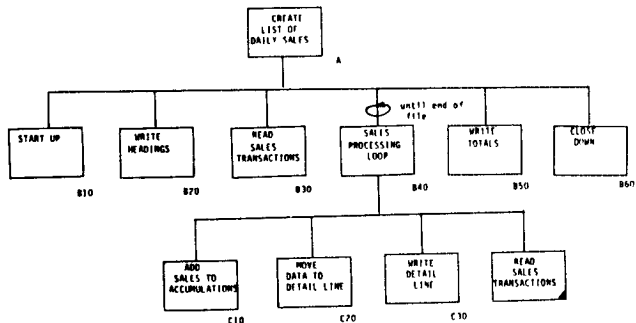


Figure 7 Create A List of Daily Sales

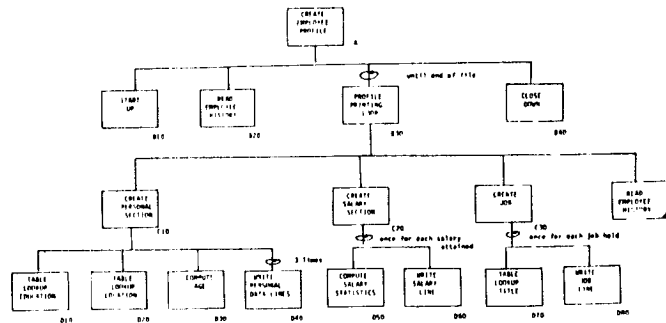


Figure 12 Create Employee Profile

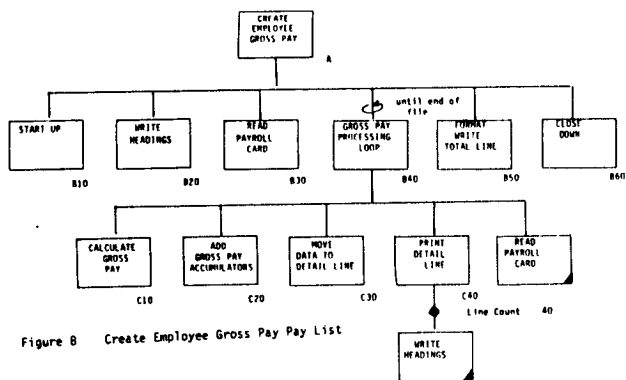


Figure 8 Create Employee Gross Pay Pay List

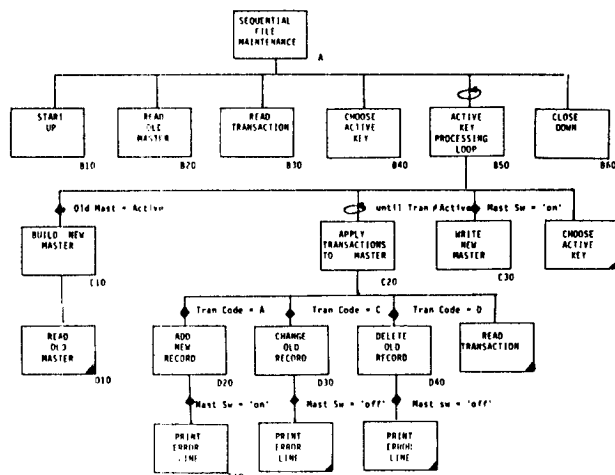


Figure 13 Sequential File Maintenance

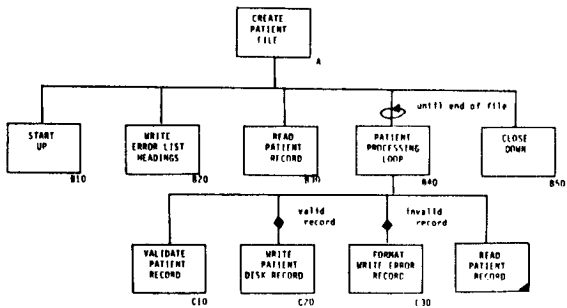


Figure 9 Create Patient File

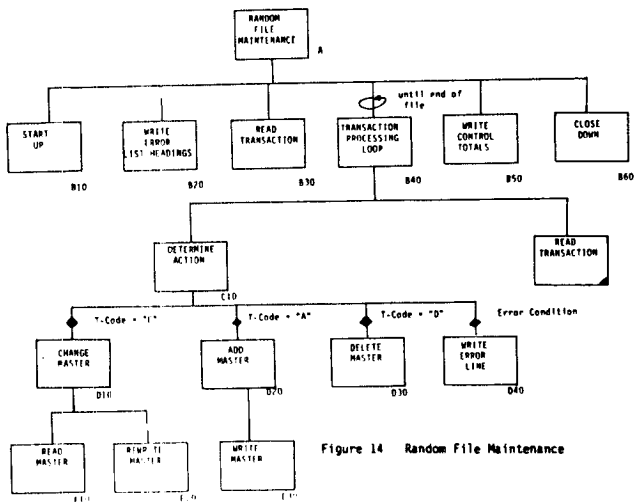


Figure 14 Random File Maintenance

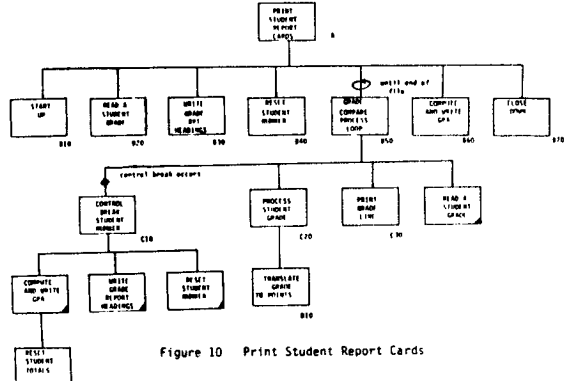


Figure 10 Print Student Report Cards

CURRICULUM FOR INTRODUCTION TO COMPUTER-BASED SYSTEMS:
A PROSPECTIVE FOR THE FUTURE

Engming Lin
Donald E. Carr
Chang-Yang Lin
Eastern Kentucky University
Richmond, Kentucky 40475

ABSTRACT

The goal of CIS-1 Introduction to Computer-based Systems in the DPMA Model Curriculum for Undergraduate Computer Information Systems Education is to prepare business majors and others to be intelligent users of computers and to understand essential elements of successful computer information systems. In order to achieve and maintain this goal the contents of the course must be revised. Required course changes result from the rapid change in computer technologies which have occurred recently and which will continue to change dramatically in the immediate future. This paper proposes an updated curriculum for the course based upon expectations for both future systems and end users.

I. INTRODUCTION

Introduction to Computer-based Systems has become one of the most popular courses at most universities and colleges. The course has three primary constituents: 1) The computer information systems (CIS) major taking the course as the entry course in the major field; 2) all business majors taking the course to partially satisfy core requirements at most business schools; and 3) other interested students taking the course to achieve a degree of computer literacy. Most students completing this course will be end-users of computerized information systems while those majoring in computer information systems will become the system developers and providers of computer information services utilized by end-users.

The DPMA Model Curriculum for Undergraduate Computer Information Systems Education, CIS-1 Introduction to Computer-based Systems is stated as a course that provides an overview of computer information systems [1]. The goal of this course, stated as STUDENT OUTCOME, is "to prepare business majors and others to be intelligent users of computers and to understand the basics of successful computer information systems..." The teaching strategy, stated as part of the course approach, "should be directed toward increasing the computer literacy needed to function in today's technological society rather than toward technical skill development." While these are factual statements, we feel the suggested contents of the course are insufficient due primarily to the dramatic changes that have occurred in computer information systems utilization in recent years. We are currently in a state of continual rapid change in the nature of computer information system design and utilization with major impact not only in the business community but in all aspects of our society. We believe that the current contents of the introductory course should undergo major change if the course is to remain relevant for students many of which will not have any additional computer education or experience prior to college matriculation and entry into their careers. The changes which we suggest will be required if the teaching strategy and the student outcomes, as stated previously, are to remain valid.

The purpose of this paper is to propose changes in curriculum for this course consistent with current overall course description and which allows for effective implementation of the stated teaching strategy to obtain goals defined in the STUDENT OUTCOME. It is obvious that the course title differs at colleges and universities throughout the country, but we shall use the title employed in the DPMA model curriculum throughout this paper.

In the next section, we discuss the current contents of the DPMA curriculum along with our view of the current general status of the many texts written for use in the course. In Section III we present our

perception of future systems and users which dictate the curriculum changes which we propose. A presentation of suggested changes for the course curriculum follows in Section IV. Finally, we attempt to draw some conclusions from this analysis.

II. CURRENT STATUS OF DPMA MODEL CURRICULUM
AND TEXTBOOKS

For more than 15 years, Introduction to Computer-based Systems has been taught as the first course in data processing (or computer information systems) and as the computer survey course in general business curricula. Topics included in CIS-1 of the DPMA model curriculum include introduction to computer information systems (10%), processing concepts (10%), input/output (10%), memory storage (10%), data communications and distributed processing (10%), computer problem solving (10%), computer programming (30%), and future of computers in society (10%). These contents were typical for this course at many colleges and universities in the late 1970s and early 1980s.

Until the late 1970s, using the computer by management or staff in most functional areas of a business firm meant receiving reports generated by the computer. The so-called end-users did not interface with the computer directly. It was considered that the end-user should possess computer literacy so that he could communicate intelligently with data processing professionals. In addition there was little consideration given directly to the growing importance of the management information systems (MIS) within the business environment. Numerous cases of disastrous implementation on computer information systems can be cited as evidence of this lack of attention in the curriculum. This was the environment for which CIS-1 of the DPMA model curriculum was originally designed. It was more of a first course in data processing than of a course oriented to future users. Most textbooks were written for the same environment. Many deans of business colleges and the American Association of Colleges and Schools of Business (AACSB), the major accrediting board, consider MIS a part of the common body of knowledge. They insist that introduction to Computer-based Systems cover the MIS topic. The DPMA Education Committee has not published an update to the model curriculum which incorporates the MIS topic into the curriculum. However, many textbooks adopted for the CIS-1 course now contain a section on MIS in their new editions.

The advancement of microcomputer technologies and use of these computers has been phenomenal since early this decade. Authors of most textbooks for Introduction to Computer-based Systems rushed to revise their books with increasing coverage of microcomputers. Some texts have even included advice on

"how to shop for a microcomputer." The emphasis appears to be switching to the end-user. Current introductory texts as well as the CIS-1 course curricula need increased emphasis on computer utilization techniques and tools which are oriented at achieving greater user productivity more rapidly.

III. THE FUTURE SYSTEMS AND USERS

In his keynote speech to the ACM84 Annual Conference held in October 1984 in San Francisco, Lewis Branscomb, Vice President and Chief Scientist of IBM, predicted that, by the end of this decade, 50% of the white-collar workforce will have workstation facilities. He also stated that the impact of a new system will be judged less on its ability to think than on its ability to "behave itself and have good manners." The message confirms the thought of many DP professionals. In its long-term plans for information systems for the 1990s, Xerox Corporation expected that most end-users in the firm will use workstations [2]. Many organizations are moving toward this same direction.

There are still a great uncertainty over the much-talked about fifth generation computer systems. Observers of the fifth generation race agree that the new computer will consist of sophisticated software that will make the use of the system more productive and easier to utilize [4]. Software as part of the 5th generation system will include knowledge-based systems and intelligent-interface systems [3]. COBOL, PL/1, BASIC and other high-level languages will become less important.

With increased workstation availability, users will obtain information they need interactively (through their fingertips on the keyboard or maybe even voice actuated) instead of waiting for computer printouts. They will use query language or graphic methods (or query by example) to retrieve information from an on-line database; most likely a relational database management system (DBMS). The workstation may be used as a terminal or as a stand-alone computer. When used as a stand-alone computer, user-friendly software packages will enable users to download required data and perform required processing locally as well as to upload and communicate in local area networks. Spreadsheet, wordprocessing, and integrated software such as LOTUS 1-2-3, or similar software will be commonly available. Decision support systems (DSS) will be available to top and middle management while expert systems will be used by professional staff. All of this software utilized directly by end-users will require more sophisticated knowledge on their part.

Managers of most business firms will still receive reports generated by the computer. But, the frequency and the volume of the reports will be reduced. Problems as well as new applications might pop up as users in every functional area of a business interface with the system everyday. This might increase the communication between DP professionals, who are the providers, and end-users.

IV. SUGGESTED CHANGES FOR THE COURSE

Based upon what are expected of the future systems and users, the three most important topics that should be added to Introduction to Computer-based Systems are 1) MIS concepts, 2) query language and database applications, and 3) microcomputer applications.

The probability that a business manager will use a high level language to write a program is much less than the probability that he will use commands of query language to retrieve data from a database designed by DP professionals. This course should provide students with hands-on experience of query language. Since users will interface with a database, they should have the general concept of a database and its functions.

The function of MIS, the role of MIS in an organization, and the users role in MIS design are important concepts which users should at least have peripheral knowledge; and, therefore, should be covered in the course. Some textbooks discuss structured design techniques and other design tools that users of MIS have rarely any chance to use. These are not what this paper suggests.

During the past 3 years, most authors have updated their textbooks for Introduction to Computer-based Systems to include fundamentals of microcomputers. The course needs more than just fundamentals. It should cover wordprocessing, spreadsheets, and some business application software. Hands-on experience on these packages will be more useful than most other topics in the course.

Adding these topics, the course may require more than three semester hours to complete. One possible arrangement would be to eliminate or severely reduce the computer programming emphasis in the course. Programming could be added in a one-hour auxiliary or complementary course offering. Programming may not remain viable in the introductory course. The DPMA curriculum suggests a two semester programming sequence for the CIS majors. Requiring programming only for majors and including only a brief introduction to BASIC in the introductory course would appear to be a good solution. Many high school students have taken a computer programming course, mostly in BASIC, before they graduate. The number of such students is increasing. In the near future, we will see that a majority of students entering the Introduction to Computer-based Systems class will have already obtained some programming background. Separating computer programming from the course would be the best arrangement.

If all topics must be included in the course, some topics will have to be shortened. In the DPMA model curriculum, computer programming related topics, including computer problem solving (10%), and computer programming (30%), take 40% of the course. This could be reduced to 30% or further down to even 20%. Hardware topics such as memory storage (10%) and peripherals (10%) require in many cases more than 20% of the course. This could be reduced to 10%. Many of the current topics are nice to know but not essential to productive utilization and interfacing with computer information systems at the user level. Reduction of these topics to the "essential" could allow space and time for inclusion of the more vital topics.

V. CONCLUSION

Nine out of ten students in the Introduction to Computer-based Systems classes will be information system users in the future. The proposed curriculum with three additional topics for the course is based on our projection of users' needs in the immediate future. The course, designed with the users' best interest in mind, should serve them better than the current curriculum.

The proposed curriculum will also serve well for CIS majors who will be the Data Processing and Information System professionals in the future. As providers of services to end users, the DP or IS professional must know and be able to predict the needs of end users. An early appreciation of these needs within the curricula will allow the CIS student to focus more rapidly in the advanced CIS courses.

REFERENCE

1. DPMA Model Curriculum for Undergraduate Computer Information Systems Education, prepared by the DPMA Education Foundation Committee on Curriculum Development, 1981.
2. Benjamin, Robert I., "Information Technology in the 1990's: A Long Range Planning Scienario", MIS Quarterly, June 1982.
3. Feigenbaum, Edward A. and McCorduck, Pamela: The Fifth Generation, Addison-Wesley Publishing Co., 1983.
4. Withington, Ted, "Winners and Losers in the Fifth Generation", Datamation, Dec. 1983, Vol. 29, No. 12, pp. 201-209.

AN INDUSTRIAL SURVEY OF END USER COMPUTING TOPICS

Robert L. Horton
Management Computer Systems
University of Wisconsin-Whitewater
Whitewater, WI 53190

End user computing and the information center are becoming increasingly vital concepts to industry. As a result, trained users are needed to take advantage of these facilities and technical personnel must be prepared to provide support. As this change occurs, computer curricula must undergo revisions to include course work in this area. To provide guidance for such revisions, a regional survey was made to gather industry suggestions for an end user course. The results of this survey are reported and broken down by industry type. There is also a brief description of the resulting course.

END USER COMPUTING

Concerned about the back log of user requests found in most data processing departments, IBM introduced the concept of the information center in the late '70's to make available to the users tools which would permit them to do their own computing. Much recent literature has pointed to end user computing as the wave of the future in data processing.

- Rockart and Flannery, MIT, recently concluded a paper (4) by stating, "The trends toward end user computing, however, are irreversible. There is little doubt in our minds that end user computing will be the dominant segment of information systems in most large companies by the end of this decade."
- Wasserman, UCSF, and Gutz, DEC, (5) agree stating, "Nonetheless, it can be seen that the nature of programming and programmers is certain to change, and that an increasing share of what we now term programming will be carried out by user-operators who will have tools at their disposal that permit them to interact naturally with a computer system and specify their requests."
- Duncan, Cal. State-Hayward, (2) reporting on a San Francisco area survey, found that major information system trends included
 - "Applications programming functions will move gradually from the central data processing department to end user organizations. Driving forces behind this trend are microcomputer software packages and the evolution of nonprocedural languages."
 - and
 - "Data processing personnel will assume a more consultative role in working with users in a joint effort to reduce traditional backlogs of requests."
- The entire February, 1984 issue of Data Management was devoted to this topic, including an article (6) which stated that in a random sample of DPMA members, 40.5% reported that they have an information center available and an additional 19% indicated that one would be implemented this year.
- Computerworld (7) recently reported that 2/3 of the organizations sampled indicated that a majority of their professional workers would have access to a workstation or terminal by the end of 1985.
- Cowan (1) and Harrar (3) have reported on the increasing number of companies that have established information centers and have been successful in saving money and increasing productivity.
- Data Processing Digest recognizes End User Computing as a category for journal articles.

Indeed, journal articles are no longer debating the acceptability of such a facility but are concerned with management, user training and software selection issues.

A COURSE IS BORN

From its inception, the Management Computer Systems (MCS) program at UW-Whitewater has relied heavily upon its Executive Advisory Board for curriculum guidance. This board consists of twenty-two data processing professionals chosen from among the region's leading industrial concerns and is charged with establishing proper goals and directions for the program as well as insuring that industry practices and standards are adhered to in the classroom.

During the spring of 1983, this board strongly suggested that a course in end user computing be implemented in the major. Many of the companies had planned or were planning to initiate information center projects that would allow non-technical personnel to satisfy some of their own data processing needs. The industry representatives agreed that this was the next major trend in the information systems field and thought it to be vital that students receive training and hands on experience in this area. Suggested topics included statistical, word processing, graphics and spreadsheet packages; microcomputer technology; 4th generation languages and query system; distributed processing; networking; and management techniques.

A new senior level course, entitled Topics in Computing, was developed to meet this need and a regional industrial survey was conducted to further determine suitable topics, software and project assignments.

SURVEY

During the winter of 1983-84, a survey was sent to 125 major companies in Wisconsin, eastern Minnesota and northern Illinois to gather suggestions for the new course. Forty-nine completed forms were returned, a response rate of almost 40%.

The survey form listed eleven suggested topics to be included in the proposed course. Responders were asked to indicate the percentage of time to be spent on each topic as well as to rate the topics on a scale of 1 (low) to 5 (high) indicating their importance to both initial job procurement and future advancement. On a similar scale, the importance of the course in the curriculum was also rated.

TOTAL RESPONSES

Table I at the end of the report shows the ranking of each topic (1-11) and, in parentheses, the average response for each of the three areas mentioned above. The topics are listed in order of ranked importance for initial job procurement. To facilitate comparisons, the topics will continue to be listed in this order throughout the report, even though the rankings may vary.

Microcomputers and fourth generation languages were considered the most important in every category followed by distributed processing, financial modeling, query systems, statistical packages, networking and graphics in varying orders. Word processing, information center and facility management seemed to be of lesser importance. The course itself received a favorable rating.

COURSE DESCRIPTION

It is interesting to note that the ratings for future advancement are significantly higher than for initial job seeking in every category. From our perspective, the initial job ranking for graphics was surprisingly low, perhaps indicating that most companies do not incorporate graphics into their routine reports. Moreover, graphics was the only topic for which every responder would allocate some portion of course-time.

Many additional topics were suggested, with database and user education being the only ones listed more than once.

The survey also contained two open-ended questions concerning student assignments and suggested software. The assignment responses varied widely from constructing a system using the various components covered in the course to simply giving the students a "feel" for each package. Most of the major software packages were recommended with many responders indicating that the specific software used was not crucial as long as it contained the usual, standard features.

The data was also analyzed by industry category.

INDUSTRY CATEGORIES

To achieve significantly sized industry categories, the responding companies were divided into the following three broad classifications:

- A - Manufacturing (27 responses)
- B - Sales/Service/Utilities (13 responses)
- C - Financial/Insurance (9 responses)

Table II contains the data in a similar format to Table I for each of the above industry categories.

For the initial job, statistics and graphics were ranked much higher by the financial/insurance companies than by the companies in the other two categories. The opposite is true of fourth generation languages. Query systems was ranked significantly higher by the service/sales/utility companies. Manufacturers gave distributed processing and networking higher rankings than did the other groups.

The data was also broken down by staff size but, unfortunately, there was a close correlation between staff size and industry type in the sample and so no new information patterns were obtained. The financial/insurance companies tended to have the larger staffs, followed by sales/service/utilities. The smallest staff sizes were typical of the manufacturing firms.

Topics in Computer Applications (950-451) is the UW-Whitewater course concerned with this survey. The course has been taught since Fall, 1983, and currently includes mainframe work with a statistical package and a database query language as well as microcomputer graphics, spreadsheet and distributed file management applications. Networking topics are also discussed. This is all covered from an information center perspective. Additional networking topics and distributed processing applications are to be included soon.

REFERENCES

1. Cowan, W. M., The "I Center" - An Office Resource Comes of Age. Office Administration and Automation, Volume 45, February, 1984, pp. 30, 31, 52.
2. Duncan, D. G., Great Expectations For Information Systems Curriculum: A Survey of MIS Management, Interface, Volume 6, Issue 2, Summer, 1984, pp. 14-17.
3. Harrar, G., Information Center, Computerworld, Volume XVII, Number 52/Volume XVIII, Number 1, December 26, 1983/January 2, 1984, pp. 71-74.
4. Rockart, J. F. and L. S. Flannery, The Management of End User Computing, Communications of the ACM, Volume 26, Number 10, October 1983, pp. 776-784.
5. Wasserman, A. I. and S. Gutz, The Future of Programming, Communications of the ACM, Volume 25, Number 3, March 1982, pp. 196-205.
6. "DPMA Members Feel the Sudden Impact of Info Centers", Data Management, Volume 22, Number 2, February 1984, p. 22.
7. "The Impact of A0", Computerworld - Office Automation, Volume 18, Number 41A, October 10, 1984, p. 16.

TABLE I - TOTAL RESPONSES

TOPICS	INITIAL JOB (1-5)		FUTURE (1-5)		TIME % (1-100)	
	Rank	Rate	Rank	Rate	Rank	Rate
Microcomputers	1	(3.16)	1	(4.06)	1	(16.29)
Fourth Generation Languages	2	(2.77)	2	(3.64)	2	(12.39)
Distributed Processing	3	(2.74)	5,6,7	(3.40)	8	(7.50)
Financial Modeling	4,5	(2.72)	3,4	(3.42)	3	(9.95)
Query Systems	4,5	(2.72)	5,6,7	(3.40)	5	(8.72)
Statistical Packages	6	(2.61)	9	(3.17)	7	(8.11)
Networking	7	(2.57)	5,6,7	(3.40)	9	(7.11)
Graphics	8	(2.52)	3,4	(3.42)	4	(8.97)
Word Processing	9	(2.40)	10	(2.86)	10	(6.39)
Information Center	10	(2.39)	8	(3.34)	6	(8.47)
Facility Management	11	(1.86)	11	(2.57)	11	(3.79)
Course Importance		3.47		4.05		

TABLE II - INDUSTRY CATEGORIES

TOPIC	INITIAL JOB (1-5)		FUTURE (1-5)		TIME % (0-100)	
	Rank	Rate	Rank	Rate	Rank	Rate
	Microcomputers	A 1 B 1 C 2	(3.30) (3.07) (2.88)	1 1 2	(4.20) (3.84) (4.00)	1 1 1
Fourth Generation Languages	A 2 B 3 C 9	(2.88) (2.92) (2.22)	2 4,5 6,7	(3.91) (3.23) (3.50)	2 2 2	(11.98) (12.72) (13.25)
Distributed Processing	A 3 B 5 C 6	(2.84) (2.69) (2.50)	3 6 8	(3.56) (3.07) (3.42)	8 8 10	(7.36) (8.00) (7.25)
Financial Modeling	A 5 B 4 C 3	(2.65) (2.84) (2.75)	8 3 3,4	(3.28) (3.46) (3.85)	3 3 6	(9.98) (10.50) (9.12)
Query Systems	A 7 B 2 C 5	(2.61) (3.00) (2.62)	6 2 10	(3.40) (3.53) (3.14)	6 4 8	(8.12) (10.45) (8.25)
Statistical Packages	A 9,10 B 6,7 C 1	(2.50) (2.46) (3.25)	9 8,9 5	(3.15) (2.92) (3.71)	7 6 9	(8.10) (8.31) (7.87)
Networking	A 4 B 6,7 C 8	(2.70) (2.46) (2.37)	7 4,5 3,4	(3.36) (3.23) (3.85)	10 5 4	(5.72) (8.45) (9.62)
Graphics	A 8 B 8 C 4	(2.57) (2.30) (2.66)	5 8,9 1	(3.42) (2.92) (4.25)	4 9 3	(9.54) (6.86) (10.12)
Word Processing	A 6 B 9,10 C 10	(2.64) (2.15) (2.11)	10 10 9	(2.92) (2.53) (3.25)	9 11 5	(6.36) (4.22) (9.50)
Information Center	A 9,10 B 9,10 C 7	(2.50) (2.15) (2.42)	4 7 6,7	(3.50) (3.00) (3.50)	5 7 7	(8.52) (8.18) (8.75)
Facility Management	A 11 B 11 C 11	(2.22) (1.46) (1.50)	11 11 11	(2.01) (2.23) (2.42)	11 10 11	(3.62) (4.27) (3.62)
Course Importance	A B C	3.39 3.45 3.75		4.13 3.90 4.00		

As an aid to reading this table note that in the Initial Job category, the Microcomputer topic is ranked first with an average rating of 3.30 by the Manufacturing companies (Group A), is ranked first with an average rating of 3.07 by the Sales/Service/Utility companies (Group B), and is ranked second with an average rating of 2.88 by the Financial/Insurance companies (Group C).

THE INFORMATION PROCESSING SYSTEM MODEL:
A CONCEPTUAL MODEL FOR CIS INSTRUCTION
James A. O'Brien, Northern Arizona University

ABSTRACT

The information processing system model is a comprehensive yet simple conceptual framework that can be used to better organize and communicate many other concepts, facts, and technological developments in the field of computer information systems. It integrates the basic information processing functions of input, processing, output, storage, and control, with the hardware, software, and people resources used in information processing. It thus provides instructors with a fundamental conceptual framework which they can teach students to use to help them: 1) understand several important, but complex concepts and technological developments that are part of the dynamic CIS area, and 2) apply such knowledge to the analysis, development, and management of computer-based information systems.

PART I: ORIGINS OF THE INFORMATION PROCESSING SYSTEM MODEL

The field of computer information systems (CIS) is full of concepts like management information systems (MIS), decision support systems (DSS) and information resource management (IRM) that are frequently criticized as "buzzwords", which are too nebulous to apply to real world situations. However, it is possible that these and several other major concepts in information processing are difficult to apply because they are based on inadequate system models of information processing itself! We need a better conceptual framework to help us teach our students how to understand and apply the many concepts, facts and technological developments in the dynamic field of computer information systems. Let's take a look at some of these models, and several recommended alternatives. But first, let's present a brief rationale for the use of the term "information processing".

DATA AND INFORMATION PROCESSING

Data processing has traditionally been defined as the processing of data to transform it into information. Thus, data processing consists of any actions which make data usable and meaningful, i.e., transforms data into information. However, the term information processing is gradually replacing the term data processing, for two major reasons:

- * Information processing is a more generic concept that covers both the traditional concept of processing numeric and alphabetic data, and the concept of word processing, in which text data, (words, phrases, sentences, paragraphs) are processed into letters, memos, reports, and other documents.
- * Information processing is a concept which emphasizes that the production of information products for users should be the focus of processing activities. It also emphasizes that the raw material resources being processed no longer consist only of numeric and alphabetic data, but newer forms such as text, images and voices.

THE SYSTEMS MODEL

The activity of information processing can be viewed as a "system". A system can be very simple and broadly defined as a group of interrelated or interacting elements. However, a more specific and appropriate concept of a system is typically used in information processing and computer technology. In this context, a system can be defined as a group of interrelated components that work toward the attainment of a common goal by accepting inputs and producing outputs in an organized transformation process. See Figure 1.

Such a system (sometimes called a dynamic system) has three basic components:

- * Input consists of elements that enter the system in order to be processed.
- * Processing involves "transformation" processes that convert input into output.
- * Output represents elements that have been produced by the transformation process.

FIGURE 1: THE ORIGIN OF THE DP SYSTEM MODEL: THE DYNAMIC SYSTEM MODEL

FIGURE 2: THE DP SYSTEM MODEL-VERSION 2: THE STORAGE FUNCTION IS ADDED

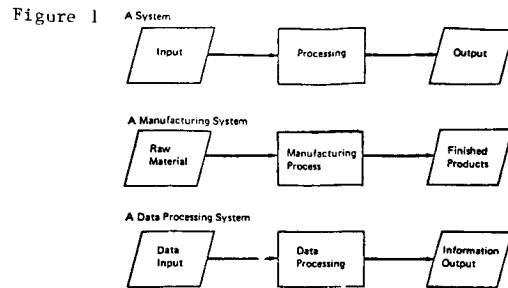


Figure 2

The systems concept is made even more useful by including two additional components: feedback and control. Figure 3 illustrates a system with feedback and control components. Such a system is sometimes called a cybernetic system, that is, a self-monitoring and self-regulating system.

FIGURE 3: THE CYBERNETIC SYSTEM MODEL

FIGURE 4: THE DP SYSTEM MODEL: VERSION 3: FEEDBACK AND CONTROL FUNCTIONS ADDED

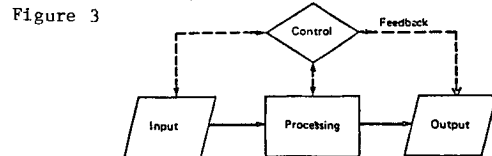
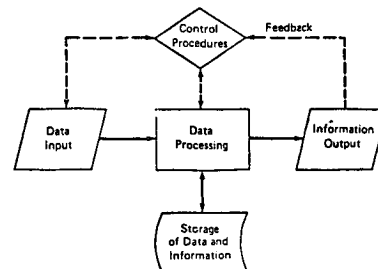


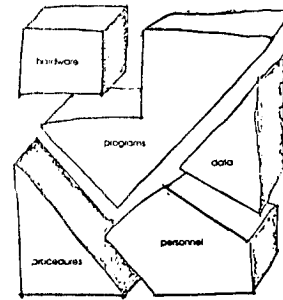
Figure 4



- * **Feedback** is information concerning the components and operations of a system.
- * **Control** is a systems component that monitors and evaluates feedback to determine whether the system is moving toward the achievement of its goal, and then makes any necessary adjustments to the input and processing components of the system to insure that proper output is produced. (Note: The feedback function is frequently included as part of the control function of a system. The responsibility of the control function is then to develop as well as monitor and evaluate feedback and make necessary adjustments to a system.)

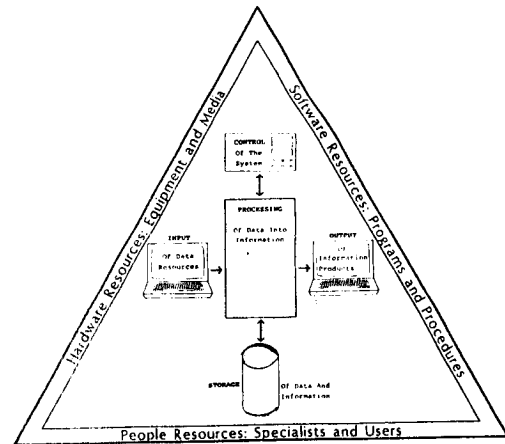
Information processing is system of input, processing, output, storage, and control functions that transform data resources into information products using hardware, software, and people as resources.

FIGURE 6: THE 5-COMPONENT MODEL OF A COMPUTER SYSTEM



This simple but comprehensive model should serve as a structural framework that ties together the many facts, concepts, and developments in the field of computers and information processing. Let's take a closer look at this model, as it is illustrated in Figure 7.

FIGURE 7: THE INFORMATION PROCESSING SYSTEM MODEL



INFORMATION PROCESSING RESOURCES

Information processing requires the use of the organizational resources of hardware, software and people to transform the data resources of an organization into information products. This is true whether we use manual methods or electronic computers for information processing.

- * **Hardware resources.** We should include in the concept of hardware resources all physical devices and materials used in information processing. Specifically, this should include not only equipment such as computers or calculators, but also all data media, that is, all tangible objects on which data is recorded, whether it be a sheet of paper or a magnetic disk.
- * **Software resources.** We should have a concept of software resources which includes all sets of information processing instructions. Specifically, this includes not only sets of computer instructions called programs, but also, the sets of information processing instructions needed by people, called procedures.
- * **People resources.** The people resources needed for information processing include both the specialists who develop and operate information processing systems, and the users who use an information processing system or the information it produces.
- * **Data resources.** Data is a very important resource to individuals and organizations. Six major types of data can be identified: (1) Traditional alphanumeric data composed of numbers and alphabetical and special characters,

TRADITIONAL DATA PROCESSING MODELS

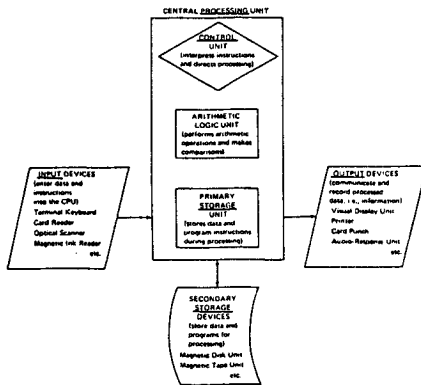
A final system component found in many data processing models is the storage function. Figures 2 and 4 illustrate these traditional models, including their use of the storage function.

- * **Storage** is the system function in which data and information are stored in an organized manner for further processing or until needed by users of the system.

THE COMPUTER SYSTEM MODEL

What is a person's first impression when he or she sees a computer? Does a microcomputer look like a combination typewriter/TV set? Does a large computer seem like a strange grouping of metal cabinets and flashing lights? Such impressions are understandable, but miss a very important concept. It is absolutely vital to anyone's effective use of computers that they understand that the computer is not a solitary electronic data processing "black box", nor is it an unrelated grouping of electronic devices performing a variety of information processing activities. We should insist that students understand the computer as a system, that is, as an interrelated grouping of components which perform the basic system functions of input, processing, output, storage, and control. Understanding of the computer as a computer system is an important capability for all computer users. For example, students should be able to visualize any computer (from a microcomputer to a supercomputer!) as a system of hardware devices organized according to the five basic system functions. See Figure 5.

FIGURE 5: A COMPUTER SYSTEM



THE 5-COMPONENT MODEL

A major attempt to develop a better model of information processing in computer-using organizations has been made by David Kronke. His 5-Component Model of a computer system consists of (1) hardware, (2) programs, (3) data, (4) procedures, and (5) personnel. His model is an improvement over previous models because it adds the new dimension of hardware, programs, and personnel to the conceptual framework of information processing systems. However, the model fails to formally incorporate the five basic information processing system functions into its framework. It also adds to the semantic confusion in information processing by being called a model of a "computer system". See Figure 6.

PART II: THE INFORMATION PROCESSING SYSTEM MODEL

Information processing should be understood in the following context:

(2) text data consisting of sentences and paragraphs used in written communications, (3) image data such as graphic shapes and figures, (4) voice data -- the human voice, (5) tactile data -- generated by touch-sensitive materials, and (6) sensor data provided by a variety of sensors used in the control of physical processes in manufacturing, military systems, space travel, etc.

INFORMATION PRODUCTS

The production of information products for users is the only reason for the existence of all information processing resources and functions. Information is provided to users in a variety of forms. Such information products include video displays, audio responses, messages, prompts, menus, forms, documents, reports, listings, etc. Managers and other users need such information products to support their decision-making and other activities.

INFORMATION PROCESSING FUNCTIONS

We outlined previously the basic system functions of input, processing, output, storage, and control. This concept can easily be applied to information processing:

- Input of data resources.
- Processing of data into information.
- Output of information products.
- Storage of data and information resources.
- Control of the information processing system.

PART III: APPLYING THE INFORMATION PROCESSING SYSTEM MODEL

Let us now look at a few brief examples of how the information processing system model can be applied to simplify and support understanding of three major concepts in CIS: 1) systems analysis and design, 2) computer-based information systems, and 3) management information systems.

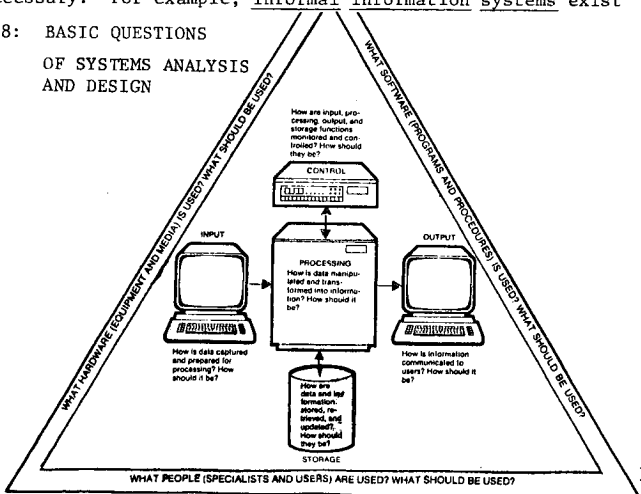
A MODEL FOR SYSTEMS ANALYSIS AND DESIGN

We should use a system's viewpoint to analyze information processing, computers, information networks, and business firms as systems of input, processing, output, storage, and control components. Developing new computer applications should thus focus on the input, processing, output, storage, and control functions of the computer-based information systems that are being proposed. Systems analysis and design then becomes a process where users and systems analysts determine how these basic information processing functions are and should be accomplished. Figure 8 is a system function diagram that illustrates this concept. It poses basic input, processing, output, storage and control questions (and related hardware, software, and people questions) that must be answered in the systems analysis and design process.

A MODEL OF COMPUTER-BASED INFORMATION SYSTEMS

What is an information system? An information system is a system that collects, transforms, and transmits information in an organization. An information system may use several kinds of information processing systems to help it provide information needed by the members of an organization. But this is not always necessary. For example, informal information systems exist

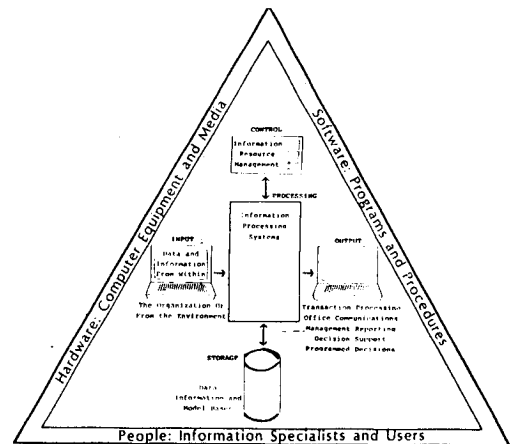
FIGURE 8: BASIC QUESTIONS OF SYSTEMS ANALYSIS AND DESIGN



in all organizations (such as the office "grapevine") which merely serve as networks to transfer information within the organization. Such information systems do not use most of the information processing system functions and activities described earlier. Therefore, we should think of information systems in a broader organizational or functional context, i.e., business information systems, management information systems, marketing information systems, etc. We can then think of information processing systems in more of a technological processing context. (For example, manual information processing systems, or electronic information processing systems).

What then is a computer-based information system? A computer-based information system can be defined as a system which uses the computer hardware, software, people, and data resources of electronic information processing systems to collect, transform, and transmit information in an organization. Figure 9 illustrates a systems model of modern computer-based information systems. Notice the following system components:

FIGURE 9: A SYSTEM MODEL OF COMPUTER-BASED INFORMATION SYSTEMS



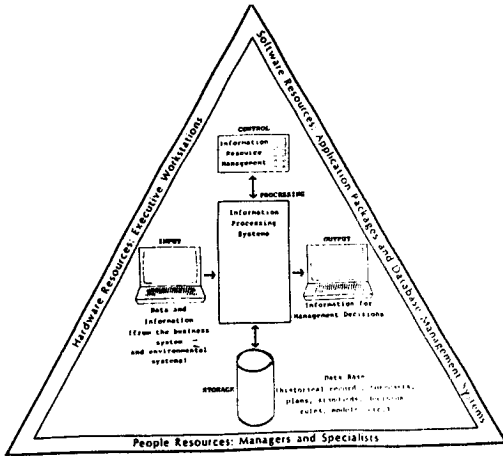
- * Input--Data and information from within the organization or from the societal environment are entered into the system.
- * Processing--Information processing systems use computer hardware (equipment and media), software (programs and procedures), personnel (information specialists and users) to transform data and information into a variety of information products.
- * Output--Transaction processing is accomplished by operational information systems.
 - Office communications is accomplished by automated office systems.
 - Management reporting is accomplished by management information systems.
 - Managerial decision support is accomplished by decision support systems.
 - Automatic programmed decision making is accomplished by programmed decision systems.
- * Storage--Data, information, and models for analysis and decision-making are stored for retrieval and processing.
- * Control--The information resource management function monitors and adjusts information system performance for optimum efficiency and effectiveness.

A MANAGEMENT INFORMATION SYSTEM MODEL

Figure 10 illustrates the management information system concept. It shows that hardware resources in the form of executive workstations, software resources of application packages and database management systems, and people resources in the form of information specialists, all help managers get the information they need from the data resources of the organization. Notice that the following system functions are included:

- * Input--Collects data generated by the other subsystems of the business system and the business environment.
- * Processing--Uses information processing systems to transform data into information products which can support managerial decision making.

FIGURE 10: A MANAGEMENT INFORMATION SYSTEM MODEL.



- * Storage--Maintains databases containing data and information in the form of historical records, forecasts, plans, decision rules, models, and other managerial and analytical techniques. Internal, external, and personal databases are used.
- * Output--Provides a wide variety of information products needed to support the decision-making activities of management (1) on demand, (2) according to a predetermined schedule, or (3) when exceptional conditions occur.
- * Control--Uses a continual process of information resource management to control the performance of the MIS, so that it efficiently and effectively meets the information needs of managers.

CONCLUSION

The information processing system model uses a comprehensive yet simple conceptual framework that can be used to better organize and communicate many other concepts, facts, and technological developments in the field of computer information systems. It integrates the basic information processing functions of input, processing, output, storage, and control, with the hardware, software, and people resources used in information processing. It thus provides instructors with a fundamental conceptual framework which they can teach students to use to help them: 1) understand several important, but complex concepts and technological developments that are part of the dynamic CIS area, and 2) apply such knowledge to the analysis, development, and management of computer-based information systems.

THE DEVELOPMENT OF AN EXPERT SYSTEM TO ASSESS
THE EFFECTS OF COMPUTER MAINTENANCE SERVICE

Avi Rushinek
Sara F. Rushinek
University of Miami, Coral Gables, Florida

ABSTRACT

This study analyzes the influence of maintenance service (MS) on overall computer user satisfaction as determined by multiple regression. The variables effectiveness, expectations, and responsiveness are found to be the most significant variables affecting overall satisfaction, whereas, the variables number of computer systems and the minicomputer (as the type of installation) have the least significant effect in the user satisfaction model. An interactive expert system (ES) for MS is designed, developed and demonstrated in this study.

INTRODUCTION TO MAINTENANCE SERVICE (MS)
RESEARCH AND THEORIES

Computers have created both problems and opportunities for managers. MS is an area of particular concern among users. In the literature, MS encompasses responsiveness and effectiveness of maintenance service.(1) Users generally assign low marks to the MS. However, these complaints that the computer industry does not provide adequate service for users have been challenged by industry.(2) This controversy intensifies as systems become increasingly more complicated and MS variables rise to prominence. Despite its growing importance, the contribution of MS to the overall satisfaction has not yet been computed. This study quantifies these contributions and summarizes MS factors of the various computer systems.

Both practitioners and theorists alike have been struggling with the various components of MS and its optimal level. Various authors have investigated the role of centralization of maintenance services as one way of providing responsiveness and effectiveness to users.(1-3) Several advantages of centralization and MS have been cited in the literature, but their exact impact upon users overall satisfaction is still in question.(4-6)

Strides have been made in assessing the user perceptions in the quality of interactive systems.(7) The applications of Artificial Intelligence (AI), expert systems, and MS have been investigated.(8-9) However, these studies have not been fully integrated and do not quantify the impact of MS on user satisfaction. Therefore, the principal objective of this study is to build upon prior theoretical work, by integrating AI, ES, MS, and measuring the impact of MS on user satisfaction.

LITERATURE REVIEW AND MAJOR OBJECTIVES

This paper describes the results of a system rating study in which the users were asked to respond to many MS questions. These questions, (independent variables) based on the literature, are considered the primary determinants of overall user satisfaction, the dependent variable. The present study relates user satisfaction to the MS variables with the use of multiple regression analysis. This analysis is the basis for the design of an expert system for forecasting user satisfaction in a specific computer installation, and then comparing it to industry standards. Information in this data base can be made available to prospective buyers of systems, but even more importantly, would remain in place to monitor user satisfaction on an on-going basis. Utilizing expert system techniques and artificial intelligence, the system can help users isolate problem areas and suggest solutions while

constantly updating the user satisfaction files through telecommunication networks.

It is important to discover which MS factors are crucial in the makeup of users' overall satisfaction toward their computer system. The literature reveals that difficulties incurred by naive managers lead to underutilization of systems, disputes over program control and ownership, and problems with data security. The lack of understanding of the system requirements is a barrier to a computer system's effective operations. MS may alleviate some of these problems with the use of expert systems.

Expert systems (ES) are specialized decision aids, which provide quantitative and qualitative analysis, probabilistic estimates along with illustrative explanations.(10) The present study explains the role of an ES as it relates to user satisfaction. Artificial intelligence (AI) facilitates the use of ES through computerized data bases, which are frequently being updated.(11-12)

Although AI has been applied to the configuration of computer systems,(13) it has not yet been applied to dealing with user satisfaction and MS variables. The stated objective of the present study is to apply ES and AI to the impact of MS on user satisfaction. Accordingly, the authors of this study introduce an expert system based on artificial intelligence (ESAI) to evaluate MS.

SURVEY METHODOLOGY AND DATA COLLECTION

This survey was based on results received from questionnaires mailed to a very carefully controlled nth sampling from specific subsets of computer users lists. A total of 15,218 questionnaires were sent to computer users. The response rate was 32%, representing 4,597 users who responded to 4,870 questionnaires. Datapro batched the remaining 4,448 valid returns by vendor, model, users, and computer types.(14)

METHODS AND PROCEDURES

A total of 179 computer systems was surveyed. The present authors coded and stored the responses to the questions (variables) on the computer (see variable legend). The data were tested for validity and consistency.

RESULTS AND DISCUSSION

Table 1 presents the statistics used for the overall test for goodness of fit for step No. 7 of the regression model. According to these tests the authors conclude that the sample multiple R of 0.66 indicates that 66% of the variation in overall

satisfaction is explained by these independent variables. Standard error of the estimate at the final step is 7.14.

TABLE 1
MULTIPLE REGRESSION

	Analysis of	Sum of	Mean
Multiple R	.66	variance	DF
R Square..	.44	Regression.	7
Adjusted R			Squares
square....	.41	Residual...	171
Standard			8724.43
error.....	7.14	Critical F	2.01 < F Value
			18.93*

* R-Square is significant at the .05 level

The relative importance of each of the predictor or independent variables on the predicted or dependent variable is described in Table 2. This relative importance is described by the BETA, the change in satisfaction, due to one standard deviation change in the predictor (criterion) variable value. These variables and their coefficients are the basis for the ESAI model, used in an interactive on-line questionnaire.

TABLE 2
VARIABLES IN THE EQUATION

Variable	B	BETA	Rank	Std. Error of B	F
Effectiveness	.357+0	.372	1	.092	15.014*
Expectations	.151+0	.260	2	.036	17.772*
System Life	-.220+0	-.145	5	.088	6.277*
Micro	.621+0	.201	4	2.053	9.154*
Responsiveness	.183+0	.205	3	.087	4.402*
Mini	.209+0	.112	6	1.218	2.934*
No. Represented	.521+0	.028	7	1.095	.226
(Constant)	.236+0				

* This F value is significant at the .05 level.

The ranking of the independent variables affecting the overall satisfaction of this ES for MS reveals some interesting results. First, it appears that the variables effectiveness, user expectations, and responsiveness (ranked 1, 2, and 3) have the strongest effects (the largest BETA values) on the dependent variable.

As can be seen in Table 2 inclusion of all seven variables has an influence on R square. This indicates that these variables should be incorporated in the model. Effectiveness has the greatest relative importance to the MS satisfaction model. In addition, if user expectations are met then user satisfaction can be enhanced.

EXPERT SYSTEMS DIAGNOSTICS FOR
MAINTENANCE SERVICE (MS) EVALUATION

Traditionally, experts have been using survey-questionnaires to evaluate MS. Such a questionnaire would usually be administered manually through an interview. In contrast, an on-line interactive data collection offers many advantages. The advantages include time saving by both the expert and the users, as well as improved precision and reliability.

Most importantly, immediate analysis and feedback becomes plausible. In fact, a computerized expert

system analyzes the data immediately after it has been entered, providing immediate feedback and diagnostics to the user. However, research and development costs, time, and efforts are major disadvantages of such ESAI systems. Thus, unless it is frequently used and updated, it will not be cost-effective.

ESAI interactively interrogates the user concerning the MS system. User responses are underlined and recorded anonymously in a data base. Subsequently, the ESAI generates the MS diagnostics audit trail. This report trails after the interactive questionnaire, providing the user with immediate feedback. Later, it may also be used by an internal or external auditor, management, or staff members for system development. This audit trail is self-explanatory. It compares the user's installation scores to industry standards based on the information from a frequently updated data base.

SUMMARY

The variables were rank ordered according to their BETA values. Effectiveness, expectations, and responsiveness were ranked the highest, while no. represented, minis, and system life ranked lower. Actually, average system life had a relatively large negative effect on the dependent variable (compared to minicomputers). This may be explained by users attributing negative feelings to the average system life in terms of their frustration of keeping up with the advances in technology, lack of control over their system, and more frequent maintenance service.

The implications of the present study are many. The overall satisfaction of computer systems can be measured by answering certain questions. The resulting information can be very useful to computer users, buyers, and sellers. Buyers can compare different computers (cross-sectional) and thus can calculate the overall satisfaction they would derive by buying a given system. This way buyers can cognitively maximize their satisfaction. Data processing managers and computer center directors can monitor, evaluate, and upgrade the MS they provide to end-users and thereby, improve user satisfaction.

REFERENCES

- (1) Linzey, H., "Computerized Centralized Maintenance: An Independent's View," *Telephony*, (November, 1983), pp. 150-155.
- (2) Harnett, J., "Who's Taking Care of The System," *Computerworld*, (August, 1983) Vol. 17, No. 31A, pp. 43-46.
- (3) Zientara, M., "Maintenance? Do It Yourself," *Computerworld*, (February, 1984) Vol. 17, No. 23A, pp. 11-12.
- (4) Adler, C., "How to Keep Automated Equipment Up and Running," *Office Administration and Automation*, (May, 1983), pp. 29-31, 82.
- (5) Gordon, R. A., "Issues in Multiple Regression," *American Journal of Sociology*, Vol. 73, (1968), pp. 592-616.
- (6) Stair, R. M., "Using In-House Computer: Some Basic Tips," *Association Management*, Vol. 33, No. 10, (October 1981), pp. 143-144.
- (7) Dzida, W., Herda, S., and Itzfeldt, D., "User-Perceived Quality of Interactive Systems," *IEEE Transactions of Software Engineering*, Vol. SE-4, No. 4, (July, 1978).
- (8) Kearsley, G., "The Relevance of AI Research to CAI," Research Report DERS-06-04-RIR-77-2, (August, 1978), pp. 1-25.

monitoring their implementation. Using such a system would eliminate the possibility of having planned a strategy package based on incomplete or erroneous environmental information due to human errors or limitations occurring in the manual process of collecting and maintaining information. Furthermore, while managers sometimes find it difficult to reason through a complicated network of operational rules to conduct simulations and analyses, the computer based intelligent system does not have limitations in this aspect of its capability.

HOW MUCH THE SYSTEM RESEMBLES OTHER A I SYSTEMS AND THE DECISION SUPPORTING SYSTEM

The intelligent strategy planning system is proposed to be built on the technologies developed in the field of artificial intelligence. Constructing a knowledge base for such a system would call for the use of A I knowledge representation technology. The system's reasoning and decision making capability depends on A I heuristic search method. A I research products in natural language dialogue provides the technological background for the system's capability to carry dialogue in natural language with the manager. However, some of these A I technologies would be applied in this system in a somewhat different manner. The knowledge base in most of A I systems is relative static; that is, the contents of knowledge base tends to stay constant and unchanged. On the contrary, the contents of the knowledge base in the intelligent strategy planning system are required to be dynamic to reflect changes of the plan's environment. Responses generated by most of A I systems to a problematic situation tend to be absolute and predictable. Due to the fact that the intelligent strategic planning system is condition oriented, its responses would be tailored according to the specific set of conditional factors found in a situation. In addition, the system is a purposeful system in which its reasoning path is dominated by an ultimate objective; that is, to develop an optimal strategy plan. In such a goal oriented system, its interactions with the user are skillfully guided for achieving this ultimate goal.

The system is intended to be used for assisting managers with formulating corporation's strategic plans. For this reason, the system is likely to be misidentified as a type of management decision supporting system. The major difference existing between these two systems is the fact that the intelligent strategic planning system does not require pre-established decision models to guide its analysis and simulation as does the management decision supporting system. (2) In the strategy development process, the intelligent system builds models for managers according to their specifications of value system and order of choices.

- (1) For a discussion of the role of the procedural rules in a knowledge base, see Winograd, T. 1972. *Understanding Natural Language*, New York: Academic Press.
- (2) For a discussion of the decision support system, see Kroeber, Donald & Watson, Hugh. 1984. *Computer-Based Information System*, New York: MacMillan Publishing Company.

BASS: BUSINESS APPLICATION AND SUPPORT SYSTEM

L. A. Adams and M. A. Bassiouni

Department of Computer Science
University of Central Florida
Orlando, Florida 32816

ABSTRACT

In this paper, we describe the design and implementation of the interface BASS (Business Application and Support System) built on top of the database management system INGRES. BASS uses the form concept and utilizes menus so that the enduser can easily construct his/her own operator. Each application is modeled and described by a set of related graph nodes. Both the interfaces to the database administrator and to the enduser are described.

INTRODUCTION

The power of today's computing technology is not being adequately utilized in most enterprises. The problem lies not with the machines themselves but in the methods used in creating applications. Traditionally, the focus of data base research was the static representation of facts by a conceptual schema and a corresponding data model. Currently, much research effort is concerned with the development of formal models which seek to homogeneously represent data, and the processing aspect of the data by application programs. Two contributing factors are :

- (1) The constituency of data base users has changed, to include more and more nonprogrammers doing their job with the help of data base systems. The majority of those new users have little or no knowledge of programming languages and data base concepts.
- (2) The realization by these users that computers can improve their decision-making capabilities and productivity.

This situation has led to the development of user friendly data base interfaces suitable for these users. Several approaches have been proposed. Mylopoulos (3) proposed the application programming language TAXIS, which allows a very high level specification of application programs by tightly integrating data and programs. Hartwick (1) proposed the integration of program libraries, offering a set of application programs, and a data base system. Studer(8) proposed an Application

Support System, integrated as a top level interface into the Distributed Data Base Management System (DBMS), FOREL. This interface facilitates the development of technical and commercial applications. Only descriptive information and "operators" (a set of application programs) need be specified.

In this paper, we will discuss the design and implementation of a 'user-friendly' interface for the Business Application and Support System(BASS), which is integrated as a "top level" interface into the the Data Base Management System, INGRES. The purpose of this interface is to:

- (1) Permit the users of data base management system to communicate with the system in a practical, efficient and easily acceptable manner.
- (2) Increase the productivity of application creation, by automating the process.
- (3) Increase the amount of data base information available to user to assist the decision making process.

The Choice of INGRES

INGRES (5) is a relational database management system designed at the University of California at Berkeley, and which runs on top of the UNIX operating system. It uses the technique of query modification to provide a quite sophisticated level of integrity constraints, views (6), and protection (7). The user interface in INGRES is provided by

the relational-calculus language QUEL. In the interactive mode, a terminal monitor process is created to allow the user to edit and execute the commands from his terminal. In the noninteractive mode, the QUEL commands are embedded into a program written in the general-purpose programming language C (resulting in the EQUEL language). The executable module of the C program replaces the interactive monitor process. Details about the INGRES system, QUEL, and EQUEL can be found in (5,6,7,9). We now give a very brief discussion of QUEL so that the reader becomes familiar with the environment on top of which BASS is built.

A QUEL interaction typically consists of one or more range variable declaration, followed by the command statement (e.g., retrieve, delete, replace, etc.). Consider the following two relations:

```
TEACHER (name,rank,sal,dept)
DEPT (dept,head,program)
```

The following interaction prints the names of all PhD granting departments along with their average salary of assistant professors.

```
range of t is TEACHER
range of d is DEPT
retrieve (dept=d.dept,Avg_Asst_Sal=
    avg(t.sal by d.dept
        where t.rank="Asst_Prof"
        and t.dept=d.dept))
where d.program="PhD"
```

The following interaction gives 10% raises for the associate professors having the salary inversion problem, i.e., those whose salary is less than the highest salary among assistant professors in their department.

```
range of t,c is TEACHER
replace c (sal=1.1*c.sal)
where c.rank="Assoc_Prof" and
    c.sal < max(t.sal by c.dept
        where t.rank="Asst_Prof"
        and t.dept=c.dept)
```

As can be seen from the above examples, QUEL is a powerful query language, but unexperienced users will have difficulty getting their task done. The interface BASS helps those users interact with INGRES in a friendly manner, similar to that described in (8) for the POREL system.

Objectives

BASS can be used for developing commercial applications, supporting a wide range of interaction between the user and the data base system using descriptive specifications.

Since BASS runs on top of INGRES, the operators are accessing the data base via the host language interface EQUEL, provided by INGRES. EQUEL is the query language QUEL embedded in the general purpose programming language, C. The relational-calculus query language QUEL is used for data definition and manipulation. BASS supports the end user during the process of creating an operator for his/her application activity. It also supports the definition of personalized application functions with fairly complex logic, and supports the end user in

specifying the input or output data for a selected operator by providing simple labeling mechanisms for interpreting the results.

These system features are based on the descriptive information specified for the applications and operators which are translated into executable code. These requirements are embedded in corresponding graph structures.

BASS offers two interfaces : (1) the Application Administrator Interface (AAI) and, (2) the End-user Interface (EI). The AAI allows the interactive specification of the Application Description Graph (ADG), the Operator Specification Graph (OSG) and operator dialogue parts by the Application Administrator (AA). The EI allows a stepwise specification of input/output data for the selected operator(s).

BASS : A DESCRIPTION

BASS is a top level interface into the DBMS, INGRES, and offers two interfaces which can be used by an end user to develop commercial applications which require interaction between a user and the data base. The AAI allows the interactive specifications of ADG, OSG, and operator dialogue parts information. The EI allows a stepwise specification of input/output data for the selected operator.

APPLICATIONS AND OPERATORS

The most general activity of an application is divided into different subactivities, each representing an alternative. When these subactivities are performed together they represent the general activity. Additionally, several relationships exists between the related subactivities.

In BASS, the applications are structured by the Application Description Graph (ADG), a generalized AND/OR graph defined in the following manner : ADG is a directed graph with four types of nodes: Compound, Disjunctive, Iterative and Expression nodes. Each node represents an application activity or related subactivity. Cycles are not allowed in ADG. A node of an ADG can have more than one predecessor nodes, i.e. one subactivity may belong to different superior activities. The information specified for each node is primarily descriptive in nature and provides the Application Administrator with the means for describing the data flow associated with a particular application model.

In BASS, the operations on an application model is defined by a graph structure, the Operator Specification Graph (OSG). The "Operators" described by the OSG, represents the realization of an application function defined in the model. Each operator consists of a computation part and a dialogue part. The computation part contains a definition of the algorithms required for performing the application. The dialogue part is a description of the input or output data for the operator. The dialogue and the computational part of an operator are transformed by the BASS system monitor into an operator transaction occurrence which when executed will perform the activity described in the OSG.

THE APPLICATION ADMINISTRATOR INTERFACE

The Application Administrator Interface (AAI) allows the interactive specification of Application Descriptive Graph (ADG's) nodes, the Operator Specification Graph (OSG) nodes and the operator dialogue parts. The following principles are incorporated in the design strategy:

- (1) BASS guarantees a uniform description of the graph structures irrespective of the Application Administrator.
- (2) The Interface assists the Application Administrator in specifying the correct syntax and semantic information for both the ADG and OSG nodes.
- (3) The Interface provides all the key words and delimiters required in ADL.
- (4) BASS offers easy access to information about the different system features.

In satisfying the first three requirements, the interface was designed using the forms concept, i.e. an ADG or ODG and its associated dialogue parts are described by inputting the information into the corresponding form. Additionally BASS offers menus and a "Help" function to provide support during conversation with the Application Administrator, and to furnish any required supplementary information.

The Application Administrator monitor screen is divided into three windows:

- (1) The STATE WINDOW indicate BASS's current operating state.
- (2) The BASS USER WINDOW contains either a form, a menu or information presented by BASS to the user in response to any user request.
- (3) The ERROR WINDOW displays all the system error messages and any information needed to correct user errors.

Each form is divided into a set of fields, into which the information is inputted by the user. The screen cursor guides the input process by sequentially tabulating from one field to another. When the information specification process is completed the user informs the AAI interface, which checks the information specified by the forms for any syntax or semantic errors. If any errors are found, then the AAI informs the user about the error(s) and guides the user in specifying the correct information.

OPERATOR INPUT/OUTPUT DATA

The operators in BASS are high level abstractions representing the structural and behavioral properties of an elementary data application. The behavioral properties of an operator, is determined by the operations that can be performed upon it. Several operations may be performed on a single object or on a group of object(s). The operation performed on the object ensure that all its semantic constraints are satisfied.

The integration of the behavioral and the structural properties of the operator provides a mechanism for manipulating different relations in the database. In BASS this mechanism is known as a 'b_skeleton'. The b_skeleton contains the structural properties of the objects being operated upon. Using the structural properties and some user inputs, the b_skeleton is transformed into a 'b_transaction'. The transformation of a b_skeleton into a b_transaction is accomplished through interactive dialogue with the user. Using the users input data the transformation process is completed.

EXECUTING OPERATORS

The execution of a BASS operator is controlled by the BASS system monitor which interprets the operator's structural description. If the operator is a simple one, the BASS monitor converses with the end user to obtain the input data which will effect the transformation of a b_skeleton into a b_transaction. Upon completion the BASS system monitor then transports the b_transaction to INGRES DBMS for execution.

A simple operator effectively represents a single transaction done by INGRES. A simple operator and a b_skeleton are the same. The AAI or end user has to simply specify their requirements in which is interpreted and executed in by the BASS monitor. If the operator is complex, the BASS system monitor interprets the structure description of the operators to determine the order in which the suboperators (simple operators) are executed.

THE ENDUSER INTERFACE

Communication between BASS and the enduser is controlled by the Enduser Interface (EI). It provides a system-guided dialogue based on menus. The enduser can elect to do one of several activities. The enduser can view the description information of any application and its associated operators defined in BASS by specifying its name i.e. all related subactivities which represent this activity. The enduser could select an operator for execution by the EI system monitor. All output data generated is interpreted and presented to the enduser. The enduser could also elect to create his/her own personalized operator.

CONCLUSION

In this paper we have presented an overview of the Business Application and Support System (BASS) implemented as a "top level" interface into the Data Base Management System, INGRES. Using ADG nodes any application can be modeled and described by a set of related nodes, the OSG represents the association between the realized application functions defined for the application model. The ability of the enduser to construct his/her own personalized (complex or simple) operator allows the naive user the ability to create his own application i.e. application development without programmers. If end users can create needed facilities directly then they can extract the information they need from

data bases, generate reports, create applications and create their own data bases to analyze data for decision making. For the expert user working applications can be created much faster and more effective.

REFERENCES

- (1) Hartwick, R. "Interactive manipulation and analysis of scientific data" Proc. 7th Annual Conf. of the Gesellschaft fur Informatik, Bonn, 1977.
- (2) Linton, M. "Implementing relational views of programs" Proc. ACM SIGSOFT/SIGPLAN Symp., 1984.
- (3) Mylopoulos, J. and Wong, H. "Some features of the TAXIS data model" Proc. Very Large Databases, 1980.
- (4) Ridjanovic, D. and Brodie, M. "Action and transaction skeletons: high level language constructs" ACM SIGPLAN Notice, Vol. 18, No. 16, 1983.
- (5) Stonebraker, M.; Wong, E. and Kreps, P. "The design and implementation of INGRES" ACM Trans. on Database Systems, Vol. 1, No. 3, 1976, pp. 190-222.
- (6) Stonebraker, M. "Implementation of integrity constraints and views by query modification" Tech. Report ERL-M514, Electronic Research Lab., University of California at Berkeley.
- (7) Stonebraker, M. "The INGRES protection system" Tech. Report ERL-M594, Electronic Research Lab, University of California at Berkeley.
- (8) Studer, R. "A dialogue interface for database applications" Proc. Very Large Databases", 1981.
- (9) Woodfill, J. and et al "INGRES version 7 reference manual" Tech Report UCB/ERL M81/61, Electronic Research Lab, University of California at Berkeley.

FUTURE COMPUTER LANGUAGES

BY

Jagtar S. Chaudhry
Research Scientist
Institute of Advanced Manufacturing Sciences
Cincinnati, OH 45209
(513) 841-8612

and

Dr. Nicklaus Damachi
Adjunct Assistant Professor
Department of Mechanical and Industrial Engineering
University of Cincinnati
Cincinnati, OH 45221
(513) 475-3863

Intended Track: Future Languages

ABSTRACT

The scope of this paper is to study some of the popular computer languages of the future such as Pascal, C, FORTH, LISP and Ada. It explains the development of these languages and dominating spheres of applications. It further evaluates the strengths and weaknesses of each language and provides an insight into determining the best language for a particular application.

Updating an old proverb, German is for talking to engineers, English is for talking to gentlemen, French is for talking to women and BASIC is for talking to Apples. The last part of this proverb is still true. But things are changing dramatically in the world of computers. New architectures are evolving, new operating systems are becoming popular and new, powerful and easy-to-use languages are flourishing. Ironically, no matter how good a language is, it takes a long time for it to become widely accepted. This is because of the lack of experienced programmers for the new languages. Also, huge libraries of software written in older languages can't be discarded overnight. There is no easy and efficient way to convert programs written in old unstructured languages into newer languages. Slowly but surely, the better computer languages are becoming more popular and widely accepted.

Table 1 shows the historical development of computer languages. Since the first high-level computer language (FORTRAN) invented by John Backus of IBM in 1956, the number of computer languages has greatly increased. Today we have around 200 distinct computer languages.

In this paper, we will look at the strengths and weaknesses of a few computer languages of the future.

PASCAL

Pascal, a highly structured language, was designed by Niklaus Wirth, a professor in Zurich, Switzerland. It was named after the French mathematician Blaise Pascal. It was designed to teach good programming skills and techniques and serves that purpose well.

Pascal programs are made up of smaller programs, each of which is a structured program. This makes large programs easy to design, implement, test, debug and maintain. Pascal programs are very easy to read and can be conveniently modified, enhanced

and understood. Pascal variables and subprograms may be named with flexibility. All Pascal variables are declared according to type at the beginning of each program or subprogram. This avoids hard-to-debug errors which cripple languages such as BASIC or FORTRAN. Pascal data types such as records, arrays, pointers and files are very flexible. There are a variety of control structures used to direct a program's order of execution. Pascal has very powerful statements such as Repeat-Until, While-Do, and Case. It was designed to be a portable language and is not well suited to access special features of its host computer. Standard Pascal does not allow dynamic arrays, though most of its newer versions include this feature. Pascal also needs more computer memory and is expensive to implement.

Table 1
Historical Development of Computer Languages

1980	ADA
	C, FORTH PASCAL
1970	APL
	BASIC, PL/1
1960	COBOL, ALGOL, LISP
1950	FORTRAN

C

C was created in Bell Laboratories in 1972. It was originally designed for in-house systems programming on Bell Lab's Unix operating system. A language by the name of B (after Bell Labs) had been developed in-house. B language never came out of Bell Labs. The authors of C, Brian Kernighan and Dennis Richie

named their language C because it followed language B.

C has become very popular for systems programming such as operating systems, compilers, word-processors and other utilities. C makes it very easy to transfer programs between computers with different processors, while still making use of the specific features of particular machines and producing efficient, compact and fast programs. C is both a high-level and low-level language. Like Pascal, it is highly structured and has very powerful statements. Like assembly language, it is very easy to manipulate bits and bytes in C. It is also called a middle-level language since it falls between assembly language and a high-level language.

C is a very compact language with only around thirty reserved keywords. This small core makes C portable and easy to implement. C is made up almost entirely of functions. Each function is stored in standard libraries. Input/output functions which tend to vary between computers are stored in such libraries, thus totally isolating the differences between various versions of C. The combination of C and the Unix operating system gives speed, portability and a comfortable working environment.

FORTH

FORTH was developed in the late 1960's by Charles Moore as his personal programming tool. When this language matured, Charles Moore was working on an IBM 1130, a "third generation" computer. He felt that this language was a leap in the fourth generation, but because the IBM 1130 would accept only five-letter identifiers, "Fourth" became "FORTH". It was designed to make the best possible use of the computer's memory and speed. This feature makes it very useful for microcomputers. Because of its speed, FORTH is used in real-time control programs and graphics. It is a standard language for sophisticated astronomical observatories. It is used to control automated movie cameras, run portable heart monitors and simulate radars for the Air Force. Today, FORTH is available on every kind of computer, from Apples to IBM mainframes.

FORTH programs consist of definitions of many words. A word in FORTH is like a procedure or subroutine. New words may be easily defined. FORTH uses reverse Polish notation for arithmetic calculations. This provides FORTH a lot of speed, but makes programs hard to read. It is an excellent language to work with under memory constraints; it has great speed and allows control and change in environments. It is less attractive to a new programmer.

LISP

LISP stands for LISt Processor and was invented by John McCarthy of Stanford in the 1960's. It is the oldest computer language that is not considered obsolete. It was designed using some of the ideas of lambda-calculus, a branch of mathematics. It is the most popular language in the field of artificial intelligence. LISP uses lists as the only data structure. A LISP program is itself a list. Programs and data are represented in the same way. This makes it easy to write programs that run other programs. LISP is very versatile in manipulating symbols. In artificial intelligence, human thoughts are represented by symbols and LISP handles them very well.

Ada

Ada was developed by the U.S. Department of Defense (DoD) in the late 1970's and early 1980's. DoD, the largest consumer of software in the world, was using many languages, which made it very cumbersome and hard to maintain and update the existing software. Since no single existing language could meet the requirements of DoD, a new language was designed and named in honor of Augusta Ada Byron, Countess of

Lovelace, daughter of Lord Byron. She is considered by many to be the first programmer as she programmed the difference engine with Charles Babbage.

Ada has a plethora of features and notational conventions which make it a very complicated language. It is criticized for having sacrificed reliability and readability in an attempt to make it powerful. Since DoD is behind Ada, it is likely to become a widely accepted language. By 1990 we can expect to see a number of hardware architectures influenced by Ada.

CONCLUSION

Any computer language can simulate any other computer language. The main points which make one language better than the other are ease, efficiency and reliability. Since different applications exploit different features of a language, one computer language may not be best suited for all applications. An all-purpose language may get very complex, like Ada. LISP is likely to dominate the field of artificial intelligence. Pascal will flourish as a general-purpose computer language. C will have a strong hold for systems programming while FORTH will do fairly well in real-time control applications.

VITAE OF AUTHORS

Jagtar S. Chaudhry is a Research Scientist at the Institute of Advanced Manufacturing Sciences, Cincinnati, Ohio. He earned his Master's Degree in Electrical and Computer Engineering in 1982 and his Master's Degree in Industrial Engineering in 1983, both from the University of Cincinnati, Ohio. He is a member of IEEE, ACM and IIE. His main field of interest is engineering, scientific and business applications of computers. He has been closely studying the present and future trends in microcomputers.

Dr. Nicklaus Damachi is an adjunct assistant professor at the University of Cincinnati. He received a Master's Degree from Ohio State University, Columbus and a Doctorate from the University of Cincinnati in Industrial Engineering. His areas of interest are software applications in the field of industrial engineering.

DEVELOPING EFFECTIVE MANAGEMENT INFORMATION END USERS

Patricia C Eiland, Auburn University at Montgomery

ABSTRACT

The selection of a decision based approach to designing a management information system is presented as a method by which effective end users can be developed. With this approach the user is required to identify his own informational requirements and support those requirements by identifying and describing the decision(s) or mandate(s) that will utilize that information. The user becomes an active participant in the system design. This should result in the increased understanding of what information can be extracted from the system, a higher level of system utilization and better end user satisfaction. End user education, involvement and ultimate satisfaction in the product produced is essential to the implementation of an effective management information system.

INTRODUCTION

A primary consideration in the development of effective end users of any management information system (MIS) is the approach taken in its design. This is true whether the MIS is to be a totally manual system, an automated system or a combination of manual and automated operations. User needs must be met if the system is to be functional. In order to best meet user needs, the MIS should be based on user decision/action relationships. This requires that the existence of any data (information) in the system be tied directly to a decision or a resultant action. The primary focal point of the design effort becomes the identification and description of all decision/action sets utilized by each end user in accomplishing their particular job functions.

DESIGN BASIS

Using this approach to MIS design, the design must be based on the premise that unless a decision is made, no information will be extracted from the MIS and unless an action has been taken as a result of that action, no information need be input back into the system.

In making a decision, a manager must have accurate and timely information available to him (her) upon which to draw conclusions and base any subsequent actions. Timing of a decision is often critical. Thus, the MIS design must reflect an ability to retrieve needed data on a timely basis. Furthermore, no manager has the time or desire to sift through endless volumes of information in order to assimilate the information that is relevant to his current decision making process. The design of the MIS must consider what information is needed, when it is needed and in what form it is needed and provide a effective means by which it can be retrieved to meet these needs.

Each time a decision is made, an action results. This action results in new information that must be fed back into the MIS in order to keep it current. Otherwise, subsequent retrieval of information to be used in making another decision will be incomplete. Thus, each action requires an input of information into the MIS which will be extracted by users in making other decisions. This cycle is repeated for all decision/action sets. This can be shown graphically as follows:

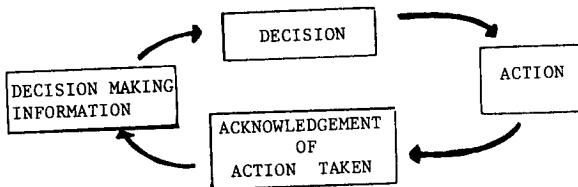


FIGURE 1

As the cycle in figure 1 implies, if no decision is made, then no action results and no information is generated or added to the MIS.

THE DESIGN TEAM

The principal design effort and responsibility usually rests on the Data Processing Staff of an organization or on a design consultant team that is contracted to develop the MIS. It is these people that must lay a foundation and create an environment that will encourage effective end use of the system. Consequently, the

design task cannot be approached from a purely technical standpoint. This approach will probably leave users with a system that does not meet their informational needs or with automation hardware and software that was not needed. Users will then circumvent the system to acquire the desired information. Soon the MIS is neither a current source of information or a tool used at all by most personnel. Manpower and financial resources have been wasted. The organization is no better off than it was prior to the system design and implementation. In fact, the negative attitudes that haunt the organization's staff after such an experience with an ill-planned, non-functional MIS will make any future efforts to provide means of improving and enhancing the decision making process at best difficult and more likely impossible. Most end users who are left with a useless MIS will shun any further interaction with any technical support staff no matter what their credentials might be! The users will usually create their own individual "mini MIS" to support their own specific informational requirements. Duplication of information as well as information gaps will soon run rampant through the organization.

In order to provide the best possible environment for preventing the problems mentioned above, there must be an extensive interaction between the technical design staff and all potential end users throughout the entire systems development and testing. This will require one-on-one interface with managers and staff on a continuous basis during the logical system design phase. The result is an organization of end users that have been actively involved in all aspects of the detail logical design of the MIS. Each user is then able to see what decisions are made at each managerial and operational level, the possible set of actions that can result from those decisions and the information needed in making those decisions. The place that each aspect of the decision making process takes in meeting the overall organization's mission becomes more evident. Each end user provides and defends his information requirements and sees how these needs will be satisfied by the MIS when it is implemented.

The only resultant information gaps in the MIS should occur when a user fails to identify such requirements. One of the primary responsibilities of the technical design staff at this point is to work closely with each user reiterating the needs established and searching for possible decision-based needs that have not come to light. My experience in systems analysis and design has shown that fewer gaps occur when this type approach is used and those occurring are more easily identified and corrected. Since the user is required to approach the design effort in terms of why each piece of information is needed and what results from having that information, redundant and useless information should for the most part be eliminated. The end user is in control of his own data requirements and how they become integrated into the MIS. The user is able to see how his data needs are incorporated into the overall informational flow of the system..

Prior to developing any "software" or physical documents, the design team must incorporate all user requirements together into a single data flow with relevant timing constraints. The user then as an opportunity to review and critique the resultant "paper design". This heavy investment of end user time will pay off in (1) the development of informed and effective end users, (2) the acquisition of an efficient and useful management tool, and (3) savings in resources required to redesign portions of the MIS when problems arise.

The systems analyst simply uses his (her) technical skills to guide users in defining and organizing system requirements into a functional processing flow that can then be translated into the MIS system processing flow and functional processing components. Up to this point little time is spent determining the medium for processing. The technical staff has been a catalyst that establishes the basic foundation upon which the MIS will be built. The analysts must then assimilate all of the requirements based on content and timing into modular specifications.

LOGICAL DESIGN STEPS

The following steps are taken in the logical design of a decision/action based MIS using the approach outlined in this paper:

- (1) Definition of the mission, goals and objectives of the organization
- (2) Identification and Description of all user decision/action events
 - + internal decision sets
 - + operations requirements
 - + externally mandated actions
- (3) Definition of all information requirements for the decision making process and mandated actions - content, presentation and timing
- (4) Accumulation of data requirements into logical processing components
- (5) Assimilation of data into a logical data hierarchy (Logical Data Base)
- (6) Finalization of logical processing components - Reporting Events and Retrieval Events
- (7) Development of the System Processing Flow

Upon completion of this process, the organization has a "paper" design of the management information system to be implemented. Any information that is to be reported or retrieved from the MIS data base is cross-referenced by the decision to which it is linked, by its position in the logical data base structure and by the system retrieval and reporting events of which it is a part. Within the system processing flow, these events are linked by content and timing.

Step 1. This step is simply a matter of defining the primary purpose and functions of the organization. This is an essential "baseline" for establishing the direction that the MIS design should take. All decisions made in the organization must in some way map back to the basic goals and objectives of the organization.

Step 2 and 3. Herein each decision that is made within the organization must be identified and described. Within each function described in Step 1 both programmatic and resource (capacity based) management decisions are made. Each decision is described in detail to show how, when and by whom the decision is made. All resultant actions must also be identified. Informational needs for the decision must be identified and a timing requirement agreed upon. Even though there are no alternatives in the actions taken as a result of external mandates, these also require retrieval of information from the MIS in order to be satisfied and may result in the need to inform the system of the action taken.

Once the data requirements have been established for every decision/action set, then the method of presentation of the information to the respective end user(s) must be defined. Even though several requirements may be identical or nearly identical in content, the organization of the information for the needed view may vary considerably based on its utilization in the specific decision making process. This factor is one of the main causes of MIS design failure. Often the format and mode of presentation of information is not considered in detail until after the physical data base has been established and the available data hierarchies defined. If the user later sees a new view that is needed or identifies a problem in an established one, it is a major re-design consideration. By identifying retrieval and reporting structures and formats "up front", the user sees in conceptual format all of the views of the information that will be available and the timing restraints that accompany each one prior to the

costly physical design and implementation of the system. Input and output document format prototypes are designed by the technical support staff and reviewed by the user until agreement is reached that their needs are met by the report. This is a prototype (pencil draft) approach. No documents are actually professionally drafted or type-set.

Step 4. This step is concerned primarily with organizing the informational requirements previously defined into logical processing components. Input and output with similar content and timing constraint as well as similar data hierarchy structure is grouped. This is the preliminary step to creating the logical MIS data base hierarchy structure and determining the basic set of processing modules required to implement the system. Figure 2 provides an example of a retrieval organization.

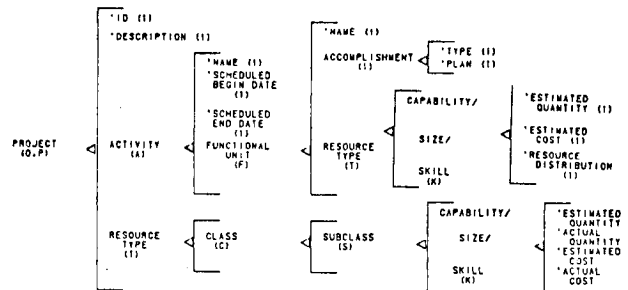


FIGURE 2

Step 5. At this point a logical data base is developed. This is accomplished by creating a "paper" document that depicts all of the hierarchical path requirements and data structures required to provide access for retrieval and reporting of information. All hierarchies established in Step 4 are reviewed and aggregated where ever possible. This will reduce duplication of data and further insure that all needs have been covered. The primary concern in this step is to insure that all the information is present that is needed in the decision making process and that all necessary paths exist in the data hierarchy organization for retrieval and updating of that information.

Step 6. It is essential from the technical design standpoint that the data structures established in step 4 and aggregated in Step 5 be associated into logical processing components. In some cases the component will consist of a single module established in Step 4. In other cases similar components may be found that can be collapsed into a single module. The input events called Reporting Events and the output events called Retrieval Events will be translated into program specifications in the physical design phase of the MIS design. A final review with end users is needed at this point to insure that the logical design for data structure and individual processing components is complete. Techniques must be developed to perform data editing and system input and retrieval functions. Figure 3 provides an overview of the logical organization of information into the MIS through the steps outlined.

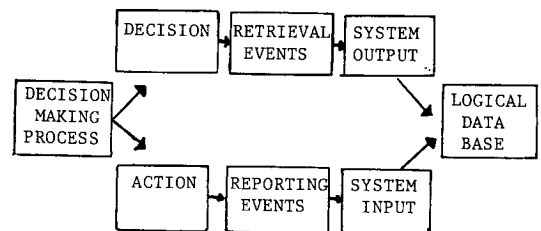


FIGURE 3

Step 7. The final step is the link between the logical and the physical design of the MIS. All requirements and procedures that have been defined heretofore must be reviewed as to the most cost effective method of processing them. This may be automated processing methods, manual methods or a combination of the two. All components must be integrated into a comprehensive processing flow. This flow must incorporate all component data structures, the required sequences of events and frequencies of occurrence into a single information flow. This finally ties everything together. Redundancies should be obvious and information gaps should be identified. The user now gets a final chance to view the sys-

tem as a whole and to see on paper how the technical design staff interprets the flow of information through the organization. Any problems that can be identified at this point can be addressed prior to the actual system design (physical design). Thus, the user has a better chance of getting what he needs and is well informed of the capabilities of the system before the costly physical design and possible hardware procurement takes place. The result is a more effective end user.

SUMMARY

In summary, the development of effective end users is both a matter of educating the user as to the capabilities of the available MIS and of insuring that the users needs have been met by the MIS design. If the design of the MIS is properly approached, it provides the key to both of these. By using a decision-based approach, users' informational needs are spelled out specifically and tied to the basis for that need through the identification of the decision action set that precipitated the inclusion of that information. Since the user is directly involved in the definition process and is required to review each step of the logical design of the system, the MIS should meet the user's needs and should give each end user a grasp onto the capabilities of the MIS in terms of his needs and the overall organization's requirements for decision making support.

REFERENCES

Powers, Michael, Adams, David and Mills, Harland. Computer Information Systems Development: Analysis and Design, Cincinnati, South-Western Publishing Co., 1984.

Thierauf, Robert. Effective Management Information Systems, Columbus, Ohio, Charles E. Merrill Publishing Co., 1984.

Warnier, J. D., Logical Construction of Programs. New York. Van Nostrand, 1974.

Yourdon, E., Managing Structured Techniques. New York. Yourdon Press, 1979.

Other references will be made available on request.

The Impact of Electronic Mail Systems
on Managerial and Organizational Communications

Mary Sumner, Assistant Professor
Management Information Systems
School of Business
Southern Illinois University at Edwardsville

Abstract

A wide range of office automation technologies, including electronic mail and personal computing, are aimed at improving the productivity of managers and professionals. Electronic mail studies reveal that these systems enable managers to create, distribute, and access information on a timely basis. In addition, these studies describe the impacts of electronic mail on the nature and volume of organizational communications, on manager/subordinate interactions, and on intradepartmental communications.

If electronic mail implementations are properly managed, the benefits may include improved organizational participation, more flexible organizational structures, and improved span of control for many managers. The payoffs may be far greater than those originally anticipated. To achieve these benefits, however, the learning curve must be encouraged, so that managers learn to use these systems effectively and begin to appreciate their real benefits.

A wide range of office technologies, including electronic mail, text processing, electronic filing, and electronic printing systems have been introduced during the past five to ten years. Most of these technologies are aimed at improving the productivity of administrative support workers and professionals in the office by making the creation, storage, production, and distribution of information more efficient.

Smith and Benjamin predict that the automated office will become the standard information handling environment for many American businesses by the end of this decade. Electronic mail systems, in particular, can improve the efficiency and effectiveness of managers by minimizing the time spent playing telephone tag and by enhancing the coordination and monitoring of staff personnel.(1)

The transfer of office technology has been rapid in many cases, and little is known about the organizational impacts of these new tools. Experience reveals that electronic mail systems do have an impact on communications patterns between managers, their subordinates and superiors. These impacts must be understood and managed in order for office automation to be successful.

The purpose of this paper is to review research findings which describe the impacts of electronic mail on managerial productivity, organizational communications, and management processes. These findings are of value to office automation professionals and to educators preparing students for roles in designing the office of the future.

IMPACTS ON MANAGERIAL PRODUCTIVITY

Since managers spend as much as 75% of their time in oral communications, electronic mail should have positive impacts on efficiency.(2) A study by Jim Bair of Bell Northern Research quantified the time savings that electronic communications systems can provide the manager as follows: 1. thirty minutes time savings per day due to reduced shadow functions (such as telephone tag); 2. sixty minutes per day resulting from optimized message handling; and 3. ten minutes time savings per day resulting from automatic preparation and distribution of messages.(3)

A thorough assessment of the impact of an electronic workstation on managerial productivity was conducted at Bell Northern Research in 1979. In this study, the communications patterns and time use of a test group of nineteen knowledge workers using an office automation system were compared with those of a control group.(4) The findings of the study showed

that the pilot system improved the users' communications, use of time, access to information, and quality of work life. Posttest results indicated a reduction in telephone use, one-on-one meetings, and interruptions for the test group. The time spent on shadow functions decreased, and these time savings appeared to be re-invested in more and better work.

A study of electronic mail at Digital Equipment Corporation by A. B. Crawford Jr., corporate manager of data processing, reported the impacts of electronic mail on communications patterns. In this study, managers reported that the electronic mail system facilitated information collection and staff coordination and enabled them to accomplish more during business hours. Soft dollar savings of between 5 percent and 15 percent were attributed to the electronic mail system.(5)

When asked about the effects of electronic mail on working relationships, many of the DEC managers stated that the amount of face-to-face contact was reduced, resulting in less personal relationships. Even though the system enabled operating managers to communicate easily with senior management, some messages channeled upward were inappropriate. In some cases, a lengthy series of messages was sent back and forth when a short meeting would have been more efficient.

In another study of electronic mail users, Ives and Olson found that managers showed an overwhelming preference for face-to-face contact over using the electronic mail system. The managers studied spent, on the average, 77 percent of their day in some form of oral contact, including meetings and brief face-to-face interruptions lasting five minutes or less.(6)

One of the reasons that managers may hesitate to use an electronic mail system, Olson suggests, is that the quality of communications is affected. Face-to-face interaction is more effective in gaining support, bargaining, or negotiating; whereas electronic mail is better for structured, task-oriented messages.(7)

IMPACT ON ORGANIZATIONAL COMMUNICATIONS

A number of studies have reported how electronic mail affects the volume and nature of communications within an organization. Studies indicate that the use of electronic mail may increase the overall volume of communications and reduce the filtering of information that occurs as messages move upwards through the corporate hierarchy. The result may be that less distorted information will be received by

higher management levels. However, this lack of filtering can also overload top management with information and, thus, become a problem.(8)

In one major corporation an electronic mail system was installed and about fifty mid and senior level executives, including the president, were given electronic mailboxes. Within the first week of installation, hundreds of messages were sent directly to the president about issues usually handled by managers reporting to him. The information overload was too great to handle, and shortly thereafter, the electronic mail system was taken out of his office. The incident points out that users of an electronic mail system may be oblivious to hierarchical communications channels.

Another area of research is the effects of electronic mail on interdepartmental communications. Traditional methods sometimes restrict communications to traditional channels, either upwards or downwards through the corporate hierarchy. However, electronic mail systems may facilitate information exchanges among persons with similar interests across departmental lines. The potential effect of this may be greater interdepartmental coordination and less rigidity in boundaries.(9) Increased electronic interactions may extend group size. Whereas a normal work group consists of eight to ten immediate superiors, subordinates, and peers, electronic mail systems may extend group size to fifty or more participants.

Another benefit of electronic mail systems is improved organizational participation. Individuals who are less outspoken at meetings or on the phone can get their ideas across using electronic mail. Increased participation in group processes, particularly in group decision-making, may create better decisions. On the other hand, open communications channels may challenge barriers that have been deliberately constructed.

To the extent that they facilitate lateral relationships and vertical communications, electronic mail systems may lead the way to more flexible and changeable organizational structures. As competitive markets change and product strategies are adapted, organizations often need to re-allocate existing resources to meet internal and external demands. The more flexible organizational structures are, the easier these changes will be.

Electronic communications networks will also make it possible to explore alternative modes of work, such as the satellite work center and work at home projects. Remote workers can obtain supervision and feedback via electronic mail that would not otherwise be possible. In one defense agency, each remote worker creates a brief message detailing the major tasks to be accomplished at the beginning of each day, and at the end of the day, notes his accomplishments. The superior can make on-line comments, suggestions, or modifications, based upon planned and actual progress. The log of electronic mail messages provides a record of individual performance, questions, problems, and feedback.

IMPACTS ON MANAGEMENT PROCESSES

Electronic mail systems also have an impact on management processes. Communications systems are used to carry out managerial functions--planning, organizing, staffing, directing, and controlling people and resources. In the course of a work day, a manager may need to request information from various subordinates, to notify them of decisions, and to provide them with specific direction. An electronic mail system can broaden the manager's span of control, by making it possible to communicate with and monitor the activities of a greater number of subordinates. Even if the manager is out of town, questions can be received and answered promptly.

In order to understand how managers use electronic mail systems to carry out their functions, frameworks for categorizing messages have been developed. Six categories, derived by inspecting messages sent by the Business Planning Group of Bell Telephone using the NLS On Line System, were developed by Leduc.(10) These categories included Personal Messages, Information Exchange, Project-Oriented, Personnel Management, and Administrative Messages.

Plain developed a new category system based upon the observed organizational function of electronic messages, including both operational and managerial communications.(11) The operational category included all routine reports sent on a regular basis. Managerial communications were split into two areas: Organizing and Problem Solving. Organizing messages consisted of requests for routine action, requests for routine information, notification of routine decisions, and personal and organizational scheduling. Problem solving messages consisted of complaints, opinions, discussion of alternatives, and notification of decisions, approvals, and solutions.

Plain's research revealed that managers undergo a learning curve. During an initial phase, the managers he studied used the electronic mail system for everything and anything they could. Over the eight month time span, Problem solving messages declined, Operational messages increased, and Organizing messages maintained a consistently high level. This finding points out the need to encourage early acceptance and continued use so that organizational learning occurs and the true benefits of electronic mail result.

In the next five to ten years, many organizations will be using electronic mail systems to improve productivity and effectiveness. However, these benefits will not come without some organizational impacts, which need to be understood. These organizational impacts will result from changes in work relationships, management processes, and even changing organizational structures.

In the planning, design and implementation of office automation technology, the role of the internal analyst becomes critical. The systems analyst must not only be able to deal with user requirements and technical considerations but also to facilitate the learning curve. The analyst can accomplish this task by spreading the experience base, providing consulting support and ongoing training, and by being sensitive to the impacts on working relationships and on interpersonal relations.

In summary, experience designing office systems indicates that technology change must be accompanied by a process of organizational learning. It is not the technology alone, but the way in which it is implemented that will determine the success of change. Once users feel that new technology is truly improving their productivity and effectiveness, they will support the organizational changes that result.

References

1. A. Smith and R.I. Benjamin, "Projecting Demand for Electronic Communications in Automated Offices," ACM Transactions on Office Information Systems, V. 1, No. 3, 1983, pp 211-229.
2. Henry Mintzberg, "The Manager's Job: Folklore and Fact," Harvard Business Review, V. 53, No. 4, July/August 1975.
3. James Bair, "Communication in the Office of the Future: Where the Real Payoff May Be," Business Communications Review, January/February 1979, pp 3-11.
4. Don Tapscott, "Investigating the Electronic Office," Datamation, March 1982.
5. A. B. Crawford Jr., "Corporate Electronic Mail-- A Communication-Intensive Application of Information Technology," Digital Equipment Corporation, May, 1982.
6. Margrethe Olson, "New Information Technology and Organizational Culture," MIS Quarterly, Special Issue 1982, pp 71-92.
7. M. Olson and H. Lucas Jr., "The Impact of Office Automation on the Organization: Some Implications for Research and Practice," Communications of the ACM, November 1982, V. 25, pp 838-847.
8. Richard G. Canning, "The Automated Office: Part I," EDP Analyzer, Vol. 16, No. 9, September 1978.
9. Jerald Hage, Michael Aiken, and Cora Marrett, "Organization Structure and Communications," American Sociological Review, V. 36, No. 5, October 1971, pp 860-871.
10. N.F. Leduc, "Communication Through Computers: Impact on a Small Business Group," Telecommunications Policy, V. 3, No. 4, September 1979, pp. 235-244.
11. H.G. Plain, "The Impact of Computer-Linked Communication on the Transmission of Regulatory and Operational Information in Organizations," Presented at the Annual Convention of the International Communications Association, Philadelphia, Pa., 1979.

ELECTRONIC WORD PROCESSING AND
OFFICE AUTOMATION: A SYSTEMS VIEW

Madjid Tavana
La Salle University

This paper examines the effect of electronic word processing on the way that offices function. At first the role of word processing in the office organization is defined, then it is suggested how to improve the operation of the office by changing its organizational characteristics to better realize the benefits of word processing technology.

Word processing is indeed only one aspect of the current trend towards office automation. The concept of an office of the future includes several other ways automating office tasks among them electronic mail, database inquiry; modeling; photo-typesetting; and teleconferencing, all linked together through a cable network. But word processing currently occupies the lead role in office automation, as it is the first computer technology to enjoy wide understanding and acceptance in the office. That lead role carries with it the role of bringing a new technology into a place where few such changes occur. The office is an anomaly of a system that it is scarcely different today than it was at the turn of this century, despite the introduction of significant new technologies like the telephone and photocopier. We can better see this static nature of the office by describing it in terms of systems concepts.

It is obvious that the typical office is an entity, both physical and conceptual, which consists of interdependent parts. And anyone who has worked in an office of any kind can confirm that the human parts of that system are capable of displaying all sorts of behavior. Whether or not it is part of a larger system, the office is largely self-controlled, so we can move on to describe the office in terms of organizational characteristics.

Contents. The parts of the office (capitalized henceforth to denote that we are referring to the typical office as a system) consist of both humans and machines.

Structure. Subgroups within the office are distinguished primarily by function. The broadest functional subgroups are executive; professional; and administrative. These three major subgroups are usually occupied by different people in the office, although there can be movement between them over time. Within the administrative subgroup are several people, e.g. typist; receptionist; personal secretary; etc.

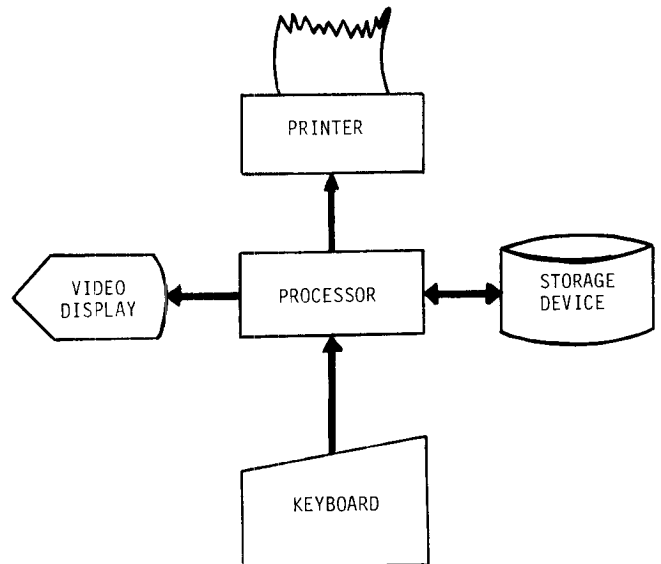
Communication. The office interaction entails much communication among its subgroups, both direct and indirect. Direct communication could take place through verbal exchanges whereby indirect communication may be indicated by indirect means, such as conflict or a lack of cooperation between subgroups.

Decision-making procedures. Some parts of the office have freedom of choice of both means and ends. Choice of ends is more limited for offices that serve larger organizations than for offices that function independently, e.g. professional firms who function entirely within an office. Choice of ends is further limited in the administrative subgroup, where the nature of the function defines its desired outcome.

The importance of these four organizational characteristics becomes clear in attempting to understand how electronic word processing is currently being used in the office.

The current technology for word processing is comprised of five components. They are as follows:

- (1) A processor with a program that executes word processing commands
- (2) A keyboard for input of commands
- (3) A video display screen that shows execution of commands
- (4) A printer for output of processed text
- (5) A storage device for work-in-process and processed text



These five components are typically configured as a workstation, containing a processor, keyboard, and display, with sole or shared access to storage and printing components. Certainly there are many other ways to configure these components, but we will deal only with this typical configuration since it is by far the most common.

To describe how these word processors are currently being used in the office, I can draw upon my own experience as well as published research that allows me to characterize my experience a 'typical' of most offices.

The word processor is introduced into the office largely through the efforts of an aggressive salesman or a highly effective advertising campaign. It joins the copier, telephone, and water cooler as one of the machines that are employed to increase the effectiveness of the human parts of the system. It is assumed from the start that the word processor will replace the typewriter in its long standing partnership with the typist subgroup of the office. By embracing that assumption, the office is true to its tradition as a static organization -- nothing about changing the office's structure, communication, or decision making process is even considered in bringing word processors into the office. The machines are simply seen as replacement for existing machines which serve the same purpose for the same people -- typing. To be sure, the word processors are usually very successful in increasing the effectiveness of typists, in two ways: 1) the machines are a vast improvement over conventional typewriters for purposes of correcting and editing text; and 2) finished documents can be made more attractive by special formatting of text. Of these two improvements, the first is seen as most significant, as it often dramatically improves the productivity of typists, as measured by finished text per unit of time.

In as much as faster typing is seen as improving the effectiveness of the office as a whole, then the word processor has improved the functioning of the office. But for most offices, the quantity of text generated does not serve organizational ends as well as does quality, where quality is measured by how effectively the textual information achieves its desired outcome. This is especially true for the types of offices which produce recommendations and proposals that are intended to motivate action and influence decisions, e.g. law firms; consulting practices; and staff departments of larger organizations. In those offices, textual material for the eyes of clients and others is more effective when it is relevant, clear, and concise -- not voluminous. Furthermore, in those offices, executives and professionals do the writing and typists do the typing -- they communicate by means of handwritten rough drafts and successive rounds of editing and retyping.

It would seem that in those types of offices word processing can do little to increase organizational effectiveness unless word processors improve the quality of writing at the executive and professional levels. However, the word processor is but a machine that makes editing faster and easier than ever, and one may ask how such a machine can improve the quality of writing in professional offices. To answer that question, we have to first revoke the common assumption that the word processor is simply a new typewriter that increases the effectiveness of typists. Removing that assumption allows us to change more than just the 'content' of the office organization. It allows us to change the structure, modes of communication, and decision-making procedures of the office in order to improve the effectiveness of written communication.

While word processing does work for some, it does not necessarily work for everyone. Even when given the choice between a personal word processor and a typist, many professionals and executives choose the typist route for reasons that are more personal than practical, e.g. they enjoy the social interaction with subordinates; or lack the interest to learn how to operate a computer keyboard, which is often seen as a clerical function.

The organizational characteristics of the office, as discussed in this paper, will play a large part in determining how quickly office technology will be understood and accepted for its potential benefits to office productivity. The scenario of word processors points out some of the same organizational barriers that will delay other concepts of the office of the future, especially electronic mail and video teleconferencing. Eroding those barriers will take a creative perspective on the part of all office personnel, and a clear distinction between what are the means and ends of the office organization.

SELECTED ANNOTATED BIBLIOGRAPHY

- Burns, Christopher J. "The Evolution of Office Information system" *Datamation*, Vol. 23, no. 4 (April 1977), pp. 60-64.
- Burns, Christopher J. "The Office in the 1980's (n.p. Arthur D. Little, Inc., n.d.), p. 24.
- Canning, Richard D., ed. "The Automated Office: Part-I" *EDP Analyzer*, Vol.16, no. 9 (September 1978), pp. 1-13.
- Canning, Richard D., ed. "The Automated Office: Part-II" *EDP Analyzer*, Vol.16, no. 10 (October 1978), pp. 1-13.
- Driscoll, James W. "People and Automated Office." *Datamation*, Vol. 25, no. 12 (November 1979), pp. 106-112.
- Holmes, Fenwicke W. "IRM: Organizing the Office of the Future." *Journal of Systems Management*, Vol. 30, no. 1 (January 1979), pp. 24-31.
- Mertes, Louis H. "Doing Your Office Over-Electronically" *Harvard Business Review* (March-April 1981), pp. 127-135.
- Morgenbrod, Horst, and Schwatzel Heinz "How New Office Technology Promotes Changing Work Methods" *Management Review*, Vol. 68, no. 7 (July 1979), pp. 42-45.
- Pappel, Harvey L. "The Automated Office Moves on" *Datamation*, Vol. 25, no. 13 (November 1979), p. 75.
- Pappel, Harvey L. "Who Needs The Office of The Future?" *Harvard Business Review* (November-December 1982), pp. 146-155.
- Rifkin, Glenn "Where Will Your Office Be" *Computer World OA* (June 15, 1983), pp. 67-74.
- Tartaglia, Benjamin W. "The Economics of Word Processing" *Journal of Systems Management*, Vol. 24, no. 11 (November 1973), pp. 8-14.
- Uttal, Bro "What's Detaining the Office of the Future" *Fortune* (May 3, 1982), pp. 176-196.
- Wayne L. Rhodes, Jr. "Office of the Future: Light Years Away?" *Infosystem*, March 1981, p. 40.
- White, Robert B. "A Prototype for the Automated Office" *Datamation*, Vol. 23, no. 4 (April 1977), pp. 83-88.

KNOWLEDGE ENGINEERING SOFTWARE SUPPORTING DELIVERY OF AN EXPERT SYSTEMS COURSE WITHIN THE CIS CURRICULUM

James H. Blaisdell
Computer Information Systems
Humboldt State University
Arcata, CA 95521

ABSTRACT

Popular acceptance by corporate functional end-users of Artificial Intelligence (AI) in the form of Expert Systems, which was spurred by the Japanese Fifth Generation Effort, has heightened interest in teaching Expert Systems within Computer Information Systems curricula. While important decisions regarding the placement and content of Expert Systems must be made, one of the most challenging is the choice of software tools to use in the delivery of Expert Systems instruction. Alternatives range from \$60,000 commercial Expert Systems development environments, which must sit atop \$40,000+ AI workstations, to locally constructed rule-based systems which will sit atop already owned or leased LISP/PROLOG environments. The field of available systems is surveyed with particular focus upon those which would be most applicable for instructional use. Finally, a recommendation regarding a home-grown PROLOG interpreter and a locally produced rule-based system are described and recommended.

BACKGROUND

Only a short while ago, Artificial Intelligence was the province of a handful of research laboratories. In the past few years several Expert Systems have emerged from the labs and entered commercial use. The announcement of the Japanese Institute for New Generation Computer Technology (ICOT) project created a surge of interest in Expert Systems.

Excited by reports in popular business and professional magazines, mesmerized by the profit potential of Expert Systems expounded upon in books like The Fifth Generation and The AI Business, functional end-users from disciplines ranging from Nuclear Engineering to Data Communications systems configuration are demanding organizational involvement in Artificial Intelligence and, in particular, Expert Systems; no wonder considering statements such as the following from The Fifth Generation.

Knowledge is information that has been pared, shaped, interpreted, selected and transformed; the artist in each of us daily picks up the raw material and makes of it a small artifact - and at the same time, a small human glory. Now we have invented machines to do that, just as we invented machines to extend our muscles and our other organs. In typical human fashion, we intend our new machines for all the usual purposes, from enhancing our lives to filling our purses. If they scourge our enemies, we wouldn't mind that either.

(Feigenbaum and McCorduck, 1983)

Questioning of end-users ranging from managers to scientists suggests that they have been reading such statements. They are not convinced that Expert Systems are the answer to all of their problems but they are convinced that they must be literate and that their firms must acquire the CIS expertise to track the developments and be prepared to implement the new technology. They earnestly believe that the Japanese Fifth Generation effort poses a serious potential threat to their competitiveness.

Questioning of several top CIS managers within major firms suggests that corporate CIS management is

either not aware of the user interest or has no idea how to develop the literacy and expertise required. Among the solutions to their problem is the development of at least a literacy level of expertise with AI by their future employees, the prospective graduates of our CIS curricula.

EXPERT SYSTEMS WITHIN THE CIS CURRICULUM

Our goals in presenting Expert Systems within the CIS curriculum are to acquaint prospective CIS graduates with the basic concepts of knowledge engineering and provide them with an opportunity to form realistic opinions regarding the opportunities and challenges which this technology presents. These goals result in an applied course with the following objectives:

- 1) A survey of existing systems focusing on the capability of such systems
- 2) An understanding of the role of knowledge, its acquisition and representation within Expert Systems
- 3) An analysis of the architectures which have been developed along with criteria for selection of an appropriate architecture
- 4) Experience with rule-based systems
- 5) The building of a proof-of-concept Expert System.

Within a CIS curriculum, coding the inference mechanism is not a primary goal.

This literacy level of understanding should be equivalent in depth to the CIS student's understanding of a topic such as Data Communications. It might be provided by a standalone special topics course. It could also be covered as a major topic within existing courses. Curriculum managers from each institution must decide upon the appropriate placement given their student and employer constituencies, faculty expertise, flexibility within the curriculum, and curricular goals.

Regardless of the placement within the curriculum, the course will be substantially more effective with

hands-on experience with the programming concepts of knowledge engineering. While few would dispute such an assertion, selecting and providing such delivery vehicles is far from straightforward or easy. The choices range from commercial development systems which sell for \$60,000 and require a minimum hardware environment the equivalent of a \$40,000 AI workstation, to a home-grown rule-based system which might run on existing IBM PCs and cost only a little keyboarding time.

GOALS OF KNOWLEDGE ENGINEERING COURSE DELIVERY SOFTWARE

The software utilized in delivering such a course then must meet the following objectives:

- 1) Provide experience with rule based systems
- 2) Demonstrate an inference mechanism architecture which can be understood by the student with a minimum exposure to the implementing language.
- 3) Provide the student with a proof-of-concept opportunity where both the difficulties of rule-based systems and the opportunities become clear.
- 4) Be delivered at a price that the institution can afford.

SURVEY OF AVAILABLE SYSTEMS

In order to facilitate the analysis of alternatives, the candidate systems have been classified within categories.

LARGE COMMERCIALY AVAILABLE DEVELOPMENT SYSTEMS

Within this category we find systems which attempt to provide state of the art knowledge engineering tools consisting of an extended knowledge acquisition/representation facility including devices such as inheritance, efficient search and pruning strategies, and metaknowledge. These systems are packaged as we have come to expect software to be marketed; with training, user documentation, software maintenance, and user support. One of the most popular, Knowledge Engineering Environment (KEE), by IntelliCorp, lists for \$60,000 and requires a minimum hardware environment to a \$40,000 AI workstation. Other systems which commercially attempt to provide a rich environment and require this type of hardware support include S-1 from Teknowledge and Loops from Xerox and art from ?????

SMALL COMMERCIALY AVAILABLE DEVELOPMENT SYSTEMS

This category is typified by small rule-based knowledge representation and a naive backward chaining inference engine. They are targeted for fully configured IBM PC compatible/comparable micros. Generally, but not always, an explanation facility and tracing facility are provided. The stripped-down support environment together with the naive search strategies make these systems of little utility for large Knowledge Engineering tasks. They are, however, very useful for learning about rule-based systems and for proof-of-concept work. The major drawback to these systems for teaching Expert Systems is relatively high cost. Systems within this category include: Texas Instrument's Personal Consultant, Teknowledge's M.1, APES, and Expert-Ease.

R&D SYSTEMS

This category of system grew from a lab environment and has usually become available to us because public money was involved in the funding of the

system. Availability of these systems is difficult to determine; documentation is frequently intended for fellow-researchers rather than end-users, and generally there is no software support or maintenance. Frequently these systems stress a specific, often experimental, architectural orientation. Initially, these systems appear to be bargains from a cost standpoint, however, they may require an expensive software support environment in which to operate. EMYCIN, for example requires, INTERLISP which lists for about \$10,000. In addition to considering hidden costs in order to support such systems, an educator acquiring them for educational use must consider that they require the user to provide his own maintenance, user support, and enhanced documentation. Systems within this category include EMYCIN, and EXPERT.

HIGHER LEVEL LANGUAGES FOR AI

The intent of tools which fit in this category is to provide a programming environment which is to Artificial Intelligence programming applications what COBOL is to business applications. An excellent example of this category is ROSIE. Developed at Rand with public money, ROSIE is available for \$200 but the earlier caution regarding underlying software interpreters applies. ROSIE requires a machine running INTERLISP. ROSIE provides English-like rules, data base facilities, and an inference engine which can work with interactive rules or make deductions from the data base (Hayes-Roth et al., 1983). Other Higher Level Languages for Artificial Intelligence include OPS5 and HEARSAY-III.

"ASSEMBLY" LEVEL LANGUAGES FOR AI

The Lingua Franca of "assembly languages" for Artificial Intelligence is LISP, an acronym for LIST Processor. PROLOG, an acronym for PROGRAMMING IN LOGIC is a strong contender as an "assembly language" for Expert Systems.

LISP is well suited to Expert Systems development since symbolic programming provides readily available knowledge representation capabilities. It has several dialects, the most popular being MacLISP (FranzLISP is a commercial adaptation of MacLISP) and INTERLISP. Common LISP is emerging as a new standard. Generally, a substantial computer is required in order to run these language interpreters. INTERLISP for example requires a LISP machine or the equivalent of a VAX. GCLISP, a Common LISP implementation for IBM PCs, requires 512K of primary storage. It provides an outstanding educational LISP environment including a strong tutorial and a GMACS editor (modeled after the MIT EMACS editor).

PROLOG is the language of choice for knowledge engineering by the Japanese for the Fifth Generation Effort. It provides a programming implementation of logic patterned after Horn clauses. It was also the language of choice by Teknowledge for implementing M-1. It is fair to point out that this is a new technology and the editing, and debugging portions of PROLOG programming environments are still not up to requirements. Various PROLOG interpreters are available for machines likely to be available to CIS departments including MPROLOG and MicroPROLOG.

OTHER SYSTEMS

Another category of systems includes tools designed to aid in development of Expert Systems by assisting either with architectural selection and development choices or with knowledge acquisition. Systems within this category include AGE, KAS, and TEIRESIAS (Hayes-Roth et al., 1983).

SUMMARY OF DELIVERY SYSTEM ALTERNATIVES

Upon initial inspection, the software required to develop a credible Expert Systems course is prohibitively expensive. In fact, however, with a little LISP and PROLOG skill and the assistance of some well-written passages regarding implementation of rule-based systems, the facilities provided by small commercial development systems can be locally produced relatively inexpensively. What is sacrificed by building your own system is: (1) prepared documentation, (2) software maintenance, and (3) user support normally provided by the vendor.

If a CIS department has no competent LISP or PROLOG programmers, buying a small commercial system offers a viable solution. The candidates listed within the "small Commercially Available Systems" category will meet the indicated goals. They cost an average of more than a thousand dollars up per copy.

A RECOMMENDED SYSTEMS

The recommended system is based upon three assumptions:

- (1) The goals of the target course are in conformance with the goals identified above.
- (2) The CIS department is not in a position to equip each student taking an expert systems course with a \$100,000 personal workstation, and funding for even a small commercially available system is marginally available.
- (3) The CIS department has access to a PC lab or a mini/mainframe running LISP on campus.

Three credible rule-based systems which meet the indicated goals can be constructed with 20-50 hours of a LISPers time. All three alternatives take advantage of the underlying "assembly" language's ability to represent knowledge.

The first one can be implemented by simply keying the LISP modules "Matching" and "Chaining" required for the forward chaining program presented in LISP (Winston and Horn, 1984). The entire program consists of less than 150 lines of code and provides a forward chaining system.

The second alternative is to code a PROLOG interpreter. By definition, the PROLOG interpreter provides an inference engine; pattern matching is provided by the PROLOG interpreter's unification algorithm. Nilsson provides a 27-line LISP implementation of a PROLOG interpreter (Campbell, 1984). It is possible to provide a minimal backward chaining experience by simply teaching students to code rules in a PROLOG data base. Unfortunately, this provides no explanation facility, no tracing, and requires learning of a subset of PROLOG. Hammond and Sergot provide suggestions for building an expert shell which will handle these tasks (Hammond and Sergot, 1983).

The third alternative is to model code from Abelson and Sussman or Charniak to produce a rule-based expert system shell (Abelson and Sussman, 1985, and Charniak et al., 1984).

In combination these three alternatives result in very credible inference engines for (1) analyzing forward and backward chaining architectures, (2) rule-based knowledge representation, and (3) proof-of-concept systems development within an introductory Expert Systems course. All of the indicated goals can be met at a minimal cost.

BIBLIOGRAPHY

- Abelson, Harold and G. Sussman with J. Sussman, (1984) Structure and Interpretation of Computer Programs, MIT Press, Cambridge, MA.
- Campbell, J. A. (Editor), (1984) Implementations of PROLOG, Ellis Horwood Limited, Chichester.
- Charniak, Eugene, C. Risbeck and D. McDermott, (1980) Artificial Intelligence Programming, Lawrence Erlbaum Associates, Hillsdale, NJ.
- Feigenbaum, Edward, and P. McCorduck, (1983) The Fifth Generation: Artificial Intelligence and Japan's Computer Challenge to the World, Addison-Wesley, Reading, MA.
- Hammond, Peter and M. Sergot, (1983) "A PROLOG Shell for Logic Based Expert Systems", Department of Computing, Imperial College, London.
- Winston, Patrick Henry, (1984) Artificial Intelligence (Second Edition), Addison-Wesley, Reading, MA.
- Winston, Patrick Henry and Berthold Klaus Paul Horn, (1984) LISP (Second Edition), Addison-Wesley, Reading, MA.
- Winston, Patrick Henry and K. Prendergast, (1984) The AI Business: Commercial Uses of Artificial Intelligence, MIT Press, Cambridge, MA.

APPENDIX A EXPERT SYSTEMS AND KNOWLEDGE ENGINEERING B&E 275

- | | | |
|------|---|--|
| WEEK | 1 | ARTIFICIAL INTELLIGENCE AND THE JAPANESE CHALLENGE TO THE WORLD. Introduction, Overview of Expert Systems, Short History of AI, Social Issues. Read pp. 1-94 TFG* |
| WEEK | 2 | Overview of Existing Expert Systems and Their Capabilities. MACSYMA, DENDRAL, MYCIN, PUFF, PROSPECTOR, CADUCEUS. Finish TFG. Read pp. 50-55 BES** plus handouts. |
| WEEK | 3 | Applications in Business XCON/XSEL, DIPMETER ANALYZER. Class Seminars, Class Presentations. Read BES 1-50 and Handouts |
| WEEK | 4 | Supporting Disciplines and Basic ES Concepts. Cognitive Psychology, Philosophy - formal logic - Predicate Calculus, Search, Problem Decomposition. Read BES 59-86 |
| WEEK | 5 | Building Expert Systems Ideal Problems, Difficult Problems Knowledge Acquisition. Project Assignments, Introduction to EXSYST |
| WEEK | 6 | Expert Systems Architectures. Forward Chaining, Introduction to FWDCHN Backward Chaining. Read BES pp. 89-126 |
| WEEK | 7 | More Expert Systems Building. Read BES pp. 127-167 |
| WEEK | 8 | Expert Systems Tools. Large Commercial Development Systems, Shells, Higher Level Languages for AI, Assembler Languages for AI, Small Commercial Systems. Read BES pp. 169-215, 283-245 |
| WEEK | 9 | Evaluating the Technology - Is It Real or Just a Bunch of Hooyey? Read BES pp. 241-280 plus Datamation Article |

WEEK 10 Walkthroughs of Class Projects

*TFG =>The Fifth Generation
**BES =>Building Expert Systems

APPENDIX B

Hardware System Vendors

Daisy Systems (workstations)
797 Kifer Rd.
Sunnyvale, CA 94086

Lisp Machine Inc. (workstations)
6033 W. Century Blvd.
Los Angeles, CA 90045
(213) 642-1116

Symbolics (workstations)
4 Cambridge Center
Cambridge, MA 02142

Tektronix (workstations)
P. O. Box 1700
Beaverton, OR 97075

Xerox Special Information Systems (workstation)
250 N. Halstead St.
P. O. Box 7018
Pasadena, CA 91109

TI (workstations)

APPENDIX C

Software Systems Vendors and Distributors

Digital Equipment Corp. (INTERLISP - originally from
USCISI)
Continental Blvd.
Merrimack, NH 03054

Expert Software International LTD. (Expert-Ease)
4Canongate Venture
5 New Street
Royal Mile
Edinburgh
EH8 8BH

Gold Hill Computers, Inc.
163 Harvard St.
Cambridge, MA 02139

Logicware (MPROLOG)
1000 Finch Ave. W. Suite 600
Toronto, Ontario, Canada, M3J 2V5
(617) 547-2393

Programming Logic Systems Inc. (MicroPROLOG)
31 Crescent Dr.
Milford, CT 06460
(203) 877-7988

Logic Based Systems (APES)
40 Beaumont Ave.
Richmond, Surrey
TW9 2HE England
Tel: 01-940 9563

IntelliCorp (KEE)
707 Laurel St.
Menlo Park, CA 94025

Xerox Special Information Systems (LOOPS)
250 N. Halstead St.
P. O. Box 7018
Pasadena, CA 91109

Teknowledge, Inc. (M.1, S.1)
525 University Avenue
Palo Alto, CA 94301

Texas Instruments Inc. (Personal Consultant)
P. O. Box 80963
Dallas, TX 75380-9063

USING A MICROCOMPUTER DBMS TO TEACH RELATIONAL DATABASE THEORY AND PRACTICE

David L. Russell, Instructor in Computer Information Systems
Western New England College

ABSTRACT

Using C.J. Date's seminal text as a guide, this paper demonstrates that a typical microcomputer database management system (dBASE II) is "relationally complete" for instructional purposes. The question of the product's tabular structure is addressed. The main relational operators of *SELECT*, *PROJECT* and *JOIN* are then demonstrated, followed by briefer demonstrations of "housekeeping" and set operators. Given relational completeness, we conclude that any institution possessing a microcomputer can teach relational database theory and practice.

This paper seeks the answers to two questions:

- (1) Can inexpensive relational database management systems ("DBMS") on microcomputers be used to teach relational database theory and practice? and
- (2) Are microcomputers relational DBMS "Relationally complete"?

It is our hypothesis that the answer is "yes" in both cases. Many institutions offering coursework in Computer Information Systems and related fields have not developed curricula emphasizing relational databases, in part due to a perceived lack of relational DBMS tools. We will demonstrate that a typical relational DBMS is indeed "relationally complete", at least for teaching purposes. Further, we will use this microcomputer DBMS to demonstrate all of the essential operations in the relational algebra, with particular attention to *SELECT*, *PROJECT* and *JOIN*. (Throughout this paper, operations in relational theory are italicized.)

This paper, rather arbitrarily, will use the relational DBMS dBASE II, a product of Ashton-Tate. This particular product was chosen because it is overwhelmingly the most popular relational microcomputer DBMS. Illustrations are from C. J. Date's text (1), which many consider to be the seminal text in this subject area.

A universally accepted definition of "relational completeness" has not yet been achieved (2), and at the introductory and intermediate teaching levels, precise definitions are not necessary. For this purpose a generalized definition will do (3). A database system is "relationally complete" if it incorporates:

- (1) A tabular data structure; and
- (2) Support of a full relational algebra, with emphasis on the special relational operators *SELECT*, *PROJECT* and *JOIN*.

Further distinctions of a "fully relational" systems (4) need not concern us here, since none exist in practice at this time (5).

If dBASE II is relationally complete then institutions can utilize it to teach relational database theory and practice. We will deal with the two portions of the definition separately.

TABULAR DATA STRUCTURE: Does dBASE II have a tabular data structure? Although the tabular structure is critical to the definition of relational database (6), Date notes a three part distinction in the structure of the system: (a) a user data model; (b) a conceptual data model; and (c) internal representation. The last does not concern us here.

dBASE II is tabular in concept but not necessarily in the user's perspective. When the user *CREATES* a *FILE*, he is presented with a vertical list of fields. This is repeated when one *LISTS STRUCTURE* (see Example 1). Two possible reasons for this choice are:

- (1) The physical screen would not deal with the

tabular presentation; and

- (2) The designers felt the user would be more attuned to the vertical listing of fields.

Most other operations are in fact tabular in the user's perspective. *DISPLAY/LIST* present data in a tabular format (within the constraints imposed by screen width), as does *REPORT FORM* within the constraints imposed when the report is generated or subsequently edited. Clearly, then, dBASE II is at least partially tabular in its user perspective. It is entirely tabular in concept.

SUPPORT OF THE RELATIONAL ALGEBRA: Date's (7) presentation of the operational aspects of relational database begins with three "housekeeping" operations: *INSERT*, *DELETE* and *UPDATE*. All are fully implemented in dBASE II. For examples of each, see Example 2.

INSERT: Placing new rows (tuples) into a given table (relation) can be achieved in dBASE II's *INSERT* command places new information into a relation at a specific location. *APPEND* does so at the end of the relation. Since the ordering of tuples in a relation has no meaning on the theoretical plane, *APPEND* is thus a form of insertion.

DELETE: This is implemented in the dBASE II command *DELETE*, which can take either record numbers or logical statements as arguments.

UPDATE: This is implemented in the dBASE II command *REPLACE*. The dBASE II command *UPDATE* performs similarly on a file-wide basis.

The definition of "relationally complete" incorporates the implementation of the entire relational algebra, but with a major focus on the three "higher-level operators". Before continuing with other operations in the relational algebra, Date (8) demonstrates these "higher-level" relational operations: *SELECT*, *PROJECT*, and *JOIN*. The following examples are taken directly from the Date text as noted. See Example 3 relations used by Date.

SELECT: In dBASE II, we obtain a subset of rows, in the simplest way, by issuing an argument to a *DISPLAY/LIST* statement. Given relation S (see Example 3) Date (9) takes a subset of rows by stating:

```
SELECT S WHERE S#='S1' GIVING TEMP
```

The same thing can be done in dBASE II by stating:

```
DISPLAY ALL FOR SUPP:NO = 'S1'
```

(Note that it is necessary to change Date's attribute name 'S#' to 'SUPP:NO' since dBASE II does not permit the character '#' in field names.) Further, to be consistent with his theoretical theme, Date creates a new relation, *TEMP*. Although dBASE II does this conceptually, explicitly creating a new relation is not necessary. It may be helpful to do this explicitly using dBASE II's *COPY* statement with an argument:

```
COPY TO TEMP FOR SUPP:NO = 'S1'
```

Example 4 contains details of these operations.

PROJECT: We obtain a subset of columns in dBASE II by explicitly or implicitly specifying

the field names to be *PROJECTED*. We USE the relation TEMP just created and project the field (column) CITY. Date (10) codes this as:

```
PROJECT TEMP OVER CITY GIVING RESULT
While in dBASE II this is coded explicitly as:
DISPLAY ALL FIELDS CITY
or implicitly as:
LIST CITY
```

(Note that DISPLAY ALL and LIST are equivalent.) Again, dBASE II conceptually develops a new relation but we may want to be explicit in doing so with the following statement:

```
COPY TO RESULT FIELDS CITY
See Example 5 for details of these operations.
```

JOIN: Theoretically, we can combine or *JOIN* any number of relations provided they share a common field, or more exactly, a common domain in a stated field. More practically, we normally *JOIN* only two relations at a time. dBASE II will only permit two relations to be involved in a single *JOIN* operations. Date (11) does a simple *JOIN* which can be directly implemented in dBASE II. Date's code reads as:

```
SELECT SP WHERE S#='S1' GIVING TEMP1
JOIN TEMP1 AND P OVER P# GIVING TEMP2
PROJECT TEMP2 OVER PNAME GIVING RESULT
dBASE II implements the JOIN operation by dividing user memory into areas: PRIMARY and SECONDARY. One relation must be in each, with the user located in PRIMARY by default. The following dBASE II code will implement this JOIN by first performing a SELECT using the COPY method described above. The result will be USED and the relation P will then be SELECTed or placed into the SECONDARY area. The two will then be JOINed. The resulting relation will then have the field PART:NAME (Date's 'PNAME') projected.
```

```
USE SP
COPY TO TEMP1 FOR SUPP:NO = 'S1'
USE TEMP1
SELECT SECONDARY
USE P
JOIN TO TEMP2 FOR PART:NO = S.PART:NO
USE TEMP2
COPY TO RESULT FIELDS PART:NAME
```

See Example 6 for details of this operation.

Finally, Date (12) introduces traditional set operations. For our purposes, ti will suffice to demonstrate only *UNION*, *INTERSECTION*, *DIFFERENCE* and *EXTENDED CARTESIAN PRODUCT*. *DIVISION* will not be demonstrated in this paper but can be implemented in dBASE II.

UNION: Our objective is to derive a relation, A *UNION* B, to be comprised of all the tuples belonging to A or B or both. Following Date's example (13), let us derive the relation of all suppliers located in London (A) and all suppliers of part P1 (B). Ironically, dBASE II performs contrary to Date in that relations explicitly stated in Date for the three higher-level operators are implicit in dBASE II, as noted above. However, in these "traditional" operators, dBASE II forces us to explicitly derive stated relations that are implicit in Date. This contradiction results from dBASE II's practical orientation: the higher-level operators are performed a great deal more often than are the traditional set operators. To perform this operation in dBASE II, we code the following:

```
USE S
SELECT SECONDARY
USE SP
JOIN TO RESULT:1 FOR SUPP:NO=S.SUPP:NO
USE RESULT:1
SELECT SECONDARY
USE P
JOIN TO RESULT:2 FOR PART:NO=S.PART:NO
```

RESULT:2 is now a very large relation -- too large for practical use. A display of RESULT:2 is a good opportunity to demonstrate that

dBASE II implements an equijoin, not a natural join. We effect a union by utilizing the Boolean operator '.OR.' and projecting the needed fields:

```
COPY TO RESULT:3 ALL FIELDS SUPP:NO, SUPP:NAME,
STATUS,CITY FOR CITY = 'LONDON'
.OR. PART:NO = 'P1'
```

See Example 7 for the details of this operation.

INTERSECTION: The desired relation, A *INTERSECTION* B, contains only those tuples in A also held in B (14). Given the large relation RESULT:2 derived above, we implement intersection by simply changing the Boolean operator in the argument to '.AND.'.

```
COPY TO RESULT:4 ALL FIELDS SUPP:NO, SUPP:NAME,
STATUS,CITY FOR CITY = 'LONDON'
.AND. PART:NO = 'P1'
```

See Example 8 for the details of this operation.

DIFFERENCE: The desired relation, A *MINUS* B contains those tuples belonging to A but not to B. Again, given RESULT:2, it is simple to derive this relation by introducing the '.NOT.' Boolean operator.

```
COPY TO RESULT:5 ALL FIELDS SUPP:NO, SUPP:NAME,
STATUS,CITY FOR CITY = 'LONDON'
.AND. .NOT. PART:NO = 'P1'
```

Details of this operation are seen in Example 9.

EXTENDED CARTESIAN PRODUCT: The desired relation A *TIMES* B, is the set of all possible combinations of A and B. In dBASE II, this is achieved by reversing the normal order in which arguments to the *JOIN* command are given. The following code is an example:

```
USE S
SELECT SECONDARY
USE SP
JOIN TO RESULT:6 FOR SUPP:NO = P.SUPP:NO
(The normal JOIN command would state the SECONDARY area's field name after the quality sign.) RESULT:6 is now the combination of all supplier numbers in S concatenated to all supplier numbers in SP. For details of this operation, see Example 10.
```

CONCLUSION: We have demonstrated (a) that dBASE II has a conceptual tabular structure, and that the user sees this tabular structure using some of the display commands. Further, (b) dBASE II can implement all relational operators. Its implementation of the relational operators predominantly used in practice (*SELECT*, *PROJECT* and *JOIN*) is more fluid than those less commonly used (*UNION*, *INTERSECTION*, *DIFFERENCE* and *EXTENDED CARTESIAN PRODUCT* and *DIFFERENCE*). Nonetheless, no relational DBMS, operating under any environment, completely and smoothly performs all operations and "working around" a given system is inevitable. We submit that dBASE II is "relationally complete" for the purposes of introductory and intermediate-level instruction and can be effectively utilized in teaching relational database theory and practice. We further submit that any example or problem from introductory and intermediate-level texts can be implemented in this product.

This being the case, any institution possessing even the humblest microcomputer can use dBASE II or an equivalent product to teach relational database theory and practice.

EXAMPLE 1

```
. CREATE P
ENTER RECORD STRUCTURE AS FOLLOWS:
FIELD  NAME,TYPE,WIDTH,DECIMAL PLACES
001    PART:NO,C,2
002    PART:NAME,C,6
003    COLOR,C,6
004    WEIGHT,N,2
005    CITY,C,10
006    <RET>
INPUT DATA NOW? N
```

```
. USE P
. LIST STRUCTURE
STRUCTURE FOR FILE:  B:P      .DBF
NUMBER OF RECORDS:  00000
DATE OF LAST UPDATE: 01/01/85
PRIMARY USE DATABASE
FLD     NAME           TYPE WIDTH  DEC
001     PART:NO       C      002
002     PART:NAME     C      006
003     COLOR         C      006
004     WEIGHT        N      002
005     CITY          C      010
** TOTAL **                00027
```

EXAMPLE 2

```
. USE SP
. LIST
00001 S1 P1 300
00002 S1 P2 200
00003 S1 P3 400
00004 S2 P1 300
00005 S2 P2 400
00006 S3 P2 200
```

```
. GOTO 3
. INSERT BEFORE
RECORD # 00003
SUPP:NO :XX:
PART:NO :YY:
QTY      :999:
```

```
. LIST
00001 S1 P1 300
00002 S1 P2 200
00003 XX YY 999
00004 S1 P3 400
00005 S2 P1 300
00006 S2 P2 400
00007 S3 P2 200
```

```
. DELETE RECORD 3
00001 DELETION(S)
. PACK
PACK COMPLETE, 00006 RECORDS COPIED
```

```
. LIST
00001 S1 P1 300
00002 S1 P2 200
00003 S1 P3 400
00004 S2 P1 300
00005 S2 P2 400
00006 S3 P2 200
```

```
. REPLACE ALL QTY WITH 999 FOR PART:NO = 'P1'
00002 REPLACEMENT(S)
```

```
. LIST
00001 S1 P1 999
00002 S1 P2 200
00003 S1 P3 400
00004 S2 P1 999
00005 S2 P2 400
00006 S3 P2 200
```

EXAMPLE 3

```
. USE P
. LIST STRUCTURE
STRUCTURE FOR FILE:  B:P      .DBF
NUMBER OF RECORDS:  00004
DATE OF LAST UPDATE: 01/01/85
PRIMARY USE DATABASE
FLD     NAME           TYPE WIDTH  DEC
001     PART:NO       C      002
002     PART:NAME     C      006
003     COLOR         C      006
004     WEIGHT        N      002
005     CITY          C      010
** TOTAL **                00027
. LIST
00001 P1 NUT      RED      12 LONDON
00002 P2 BOLT    GREEN    17 PARIS
00003 P3 SCREW   BLUE     17 ROME
00004 P4 SCREW   RED      14 LONDON
```

```
. USE S
. LIST STRUCTURE
STRUCTURE FOR FILE:  B:S      .DBF
NUMBER OF RECORDS:  00003
DATE OF LAST UPDATE: 11/13/84
PRIMARY USE DATABASE
FLD     NAME           TYPE WIDTH  DEC
001     SUPP:NO       C      002
002     SUPP:NAME     C      006
003     STATUS        N      002
004     CITY          C      008
** TOTAL **                00019
. LIST
00001 S1 SMITH    20 LONDON
00002 S2 JONES    10 PARIS
00003 S3 BLAKE    30 PARIS
```

```
. USE SP
. LIST STRUCTURE
STRUCTURE FOR FILE:  B:SP     .DBF
NUMBER OF RECORDS:  00006
DATE OF LAST UPDATE: 01/01/85
PRIMARY USE DATABASE
FLD     NAME           TYPE WIDTH  DEC
001     SUPP:NO       C      002
002     PART:NO       C      002
003     QTY           N      003
** TOTAL **                00008
```

```
. LIST
00001 S1 P1 300
00002 S1 P2 200
00003 S1 P3 400
00004 S2 P1 300
00005 S2 P2 400
00006 S3 P2 200
```

EXAMPLE 4

```
. USE S
. DISPLAY ALL FOR SUPP:NO = 'S1'
00001 S1 SMITH 20 LONDON
.
. COPY TO TEMP FOR SUPP:NO = 'S1'
00001 RECORDS COPIED
. USE TEMP. LIST00001 S1 SMITH 20 LONDON
```

EXAMPLE 5

```
. USE TEMP
. DISPLAY ALL FIELDS CITY
00001 LONDON
.
. LIST CITY
00001 LONDON
. COPY TO RESULT FIELDS CITY
00001 RECORDS COPIED
.
. USE RESULT
. LIST
00001 LONDON
```

EXAMPLE 6

```
. USE SP
. COPY TO TEMP1 FOR SUPP:NO = 'S1'
00003 RECORDS COPIED
. USE TEMP1
. LIST
00001 S1 P1 300
00002 S1 P2 200
00003 S1 P3 400
. SELECT SECONDARY
. USE P
. JOIN TO TEMP2 FOR PART:NO = S.PART:NO
. USE TEMP2
. LIST
00001 S1 P1 300 P1 NUT RED 12 LONDON
00002 S1 P2 200 P2 BOLT GREEN 17 PARIS
00003 S1 P3 400 P3 SCREW BLUE 17 ROME
. COPY TO RESULT FIELDS PART:NAME
00003 RECORDS COPIED
. USE RESULT
. LIST
00001 NUT
00002 BOLT
00003 SCREW
```

EXAMPLE 7

```
. USE S
. SELECT SECONDARY
. USE SP
. JOIN TO RESULT:1 FOR SUPP:NO = S.SUPP:NO
. USE RESULT:1
. LIST
00001 S1 SMITH 20 LONDON S1 P1 300
00002 S1 SMITH 20 LONDON S1 P2 200
00003 S1 SMITH 20 LONDON S1 P3 400
00004 S2 JONES 10 PARIS S2 P1 300
00005 S2 JONES 10 PARIS S2 P2 400
00006 S3 BLAKE 30 PARIS S3 P2 200
. SELECT SECONDARY
. USE P
. JOIN TO RESULT:2 FOR PART:NO = S.PART:NO
. USE RESULT:2
00001 S1 SMITH 20 LONDON S1 P1 300 P1 NUT RED 12 LONDON
00002 S1 SMITH 20 LONDON S1 P2 200 P2 BOLT GREEN 17 PARIS
00003 S1 SMITH 20 LONDON S1 P3 400 P3 SCREW BLUE 17 ROME
00004 S2 JONES 10 PARIS S2 P1 300 P1 NUT RED 12 LONDON
00005 S2 JONES 10 PARIS S2 P2 400 P2 BOLT GREEN 17 PARIS
00006 S3 BLAKE 30 PARIS S3 P2 200 P2 BOLT GREEN 17 PARIS
.
. COPY TO RESULT:3 ALL FIELDS SUPP:NO, SUPP:NAME, ;
. STATUS, CITY FOR CITY = 'LONDON' ;
. OR. PART:NO = 'P1'
00004 RECORDS COPIED
. USE RESULT:3
. LIST
00001 S1 SMITH 20 LONDON
00002 S1 SMITH 20 LONDON
00003 S1 SMITH 20 LONDON
00004 S2 JONES 10 PARIS
```

EXAMPLE 8

```
. USE RESULT:2
.
. COPY TO RESULT:4 ALL FIELDS SUPP:NO, SUPP:NAME, ;
. STATUS, CITY FOR CITY = 'LONDON' ;
. AND. PART:NO = 'P1'
00001 RECORDS COPIED
. USE RESULT:4
. LIST
00001 S1 SMITH 20 LONDON
```

EXAMPLE 9

```
. USE RESULT:2
. COPY TO RESULT:5 ALL FIELDS SUPP:NO, SUPP:NAME, ;
. STATUS, CITY FOR CITY = 'LONDON' ;
. AND. .NOT. PART:NO = 'P1'
00002 RECORDS COPIED
. USE RESULT:5
. LIST
00001 S1 SMITH 20 LONDON
00002 S1 SMITH 20 LONDON
```

EXAMPLE 10

```
. USE S
. SELECT SECONDARY
. USE SP
. JOIN TO RESULT:6 FOR SUPP:NO = P.SUPP:NO
. USE RESULT:6
. LIST
00001 S1 SMITH 20 LONDON S1 P1 300
00002 S1 SMITH 20 LONDON S1 P2 200
00003 S1 SMITH 20 LONDON S1 P3 400
00004 S1 SMITH 20 LONDON S2 P1 300
00005 S1 SMITH 20 LONDON S2 P2 400
00006 S1 SMITH 20 LONDON S3 P2 200
00007 S2 JONES 10 PARIS S1 P1 300
00008 S2 JONES 10 PARIS S1 P2 200
00009 S2 JONES 10 PARIS S1 P3 400
00010 S2 JONES 10 PARIS S2 P1 300
00011 S2 JONES 10 PARIS S2 P2 400
00012 S2 JONES 10 PARIS S3 P2 200
00013 S3 BLAKE 30 PARIS S1 P1 300
00014 S3 BLAKE 30 PARIS S1 P2 200
00015 S3 BLAKE 30 PARIS S1 P3 400
00016 S3 BLAKE 30 PARIS S2 P1 300
00017 S3 BLAKE 30 PARIS S2 P2 400
00018 S3 BLAKE 30 PARIS S3 P2 200
```

REFERENCES

- (1) Date, C.J. An Introduction to Database Systems, 3rd. ed., Addison-Wesley, 1981
- (2) Date, C.J. An Introduction to Database Systems, Volume II, Addison-Wesley, 1983: p. 188 ff
- (3) Date, C.J., 1983: pp. 185-6; after Codd, E.F. "Relational Database: A Practical Foundation for Productivity", CACM 25, no. 2 (February 1982)
- (4) Date, 1983: pp. 188-196
- (5) Date, 1983: p. 185
- (6) Date, 1981: p. 187
- (7) Date, 1981: pp. 56-57
- (8) Date, 1981: pp. 73 ff
- (9) Date, 1981: pp. 75-76, Example 3.5.1
- (10) Date, 1981: p. 76
- (11) Date, 1981: p. 77, Example 3.5.3
- (12) Date, 1981: p. 205 ff
- (13) Date, 1981: p. 205
- (14) Date, 1981: pp. 205-206

Acknowledgements: The author wishes to acknowledge Prof. Van Court Hare of the University of Massachusetts for his guidance in developing this paper. In addition, the author acknowledges the secretarial staff of the Western New England College School of Business for their assistance.

The Impact of User Developed Applications
on Systems Development

Dr. Mary Sumner, Assistant Professor
Department of Management Information Systems
School of Business

ABSTRACT

User-driven computing has experienced tremendous growth in the 1980's. If users were equipped with application development tools, it was felt that much of the application backlog could be alleviated. However, studies of the scope and nature of user-developed applications reveal that these applications satisfy individual and departmental level information requirements, rather than lessening the traditional backlog. Without the use of fourth generation tools and information center support, many of these user requirements could not have been satisfied at all.

One of the critical roles of the information center analyst is to provide support for continued organizational learning. However, if end-user computing is to be cost-effective, policies must be established to assure continued growth and at the same time to maintain adequate controls over technology and application development.

In his text Application Development Without Programmers, James Martin argues that many of the problems of traditional systems development, including long development cycles, formal requirements specification, and formal maintenance procedures can be overcome by user-driven computing. In the environment Martin describes, users will work with systems analysts to create their own applications, using tools such as application generators to create prototypes. (1)

The organization of an information center, staffed with consultants who work with users, is a valuable approach to providing end-user computing. In corporations with well-designed data bases, Martin suggests that 70 percent of user needs can be met with query languages and report generators. Less than 10 percent of the end-user demands for new applications require formal programming in languages such as COBOL. In addition, the maintenance problem can be alleviated. (2) Report and application generators will provide tremendous productivity increases, eliminate formal specifications, and create applications that are cheap and quick to maintain.

Chet Mills, an expert in the management of end user computing, argues that the underlying objective of the information center is to enable operating level management to get the information they need to make more timely, better informed, accurate business decisions. (3) In particular, end user tools should reduce the time it takes to convert data into useful information for decision-making.

The organization of an information center can also benefit the information systems staff directly, Mill argues, by freeing I/S professionals from time-consuming maintenance tasks and enabling them to deal with larger scale systems issues. In companies where information centers have been implemented for over 12 months, Mill reports, I/S management has reported a reduction in the ad hoc application backlog. That is, many of the requests for changes and modifications in reports can be handled by the users themselves. With the information center in place, I/S management is in a better position to develop true corporate data repositories which end users can use to fulfill their specific information needs. (4)

Since Martin's text was written, the growth of end-user computing within organizations has been overwhelming. The November 1983 issue of EDP Analyzer reports this growth at one of Xerox Corporation's major business components between 1970 and 1980. (5) In 1970, end-user computing at this Xerox component was a very small amount of the 3.5 million instructions per second (MIPS) capacity. By 1980, end-user computing had grown to almost 40 percent of

the 70 MIPS capacity (an increase of 20 times over the decade) and was expected to increase to 75 percent of the total workload by 1990.

End-user computing, the EDP Analyzer report argues, in the not-distant future probably will overwhelm the information systems department. End users will demand services and support far in excess of what the I/S department can provide within budget--or even with a reasonable increase in the budget. (6) Some of the problems which will be likely to result from the rapid expansion of end-user computing are the incompatibility of hardware, software, and data; lack of procedures for documentation and data security; and increasing demands for support.

The success of the information center will inevitably lie in its successful management. In the December 1983 EDP Analyzer article "Coping with End-User Computing," several success factors are discussed. Providing users with access to corporate data, standardizing personal computers, offering training, and encouraging innovation are effective strategies. However, the report argues that a firm will be better able to cope with this new phenomenon if it identifies high-leverage uses for end-user computing--uses that will help the company improve and protect its competitive position. (7)

The purpose of this paper is to describe the scope and nature of user-developed applications, to identify their impact on managerial productivity, to assess their impact on application development productivity, and to identify guidelines that will assure continued growth and at the same time maintain adequate controls over end-user computing. Information systems professionals must not only be aware of these issues but also develop effective strategies to maximize the investment being made in user-driven computing. Data processing educators must develop curricula to train future information center professionals and end users in their respective roles in business systems development.

John Rockart and Lauren Flannery's study, "The Management of End-User Computing," provides a thorough view of users, their needs, and their applications within seven major organizations. Six types of end users were identified, including nonprogramming end users, command-level end users, end-user programmers, functional support personnel, end-user computing support personnel, and data processing programmers. The largest percentage studied were functional support personnel, sophisticated programmers supporting end users within functional areas. Because of the diversity of end users, Rockart argued that multiple software tools and differentiated training should be available to meet their needs. (8)

Of the end user applications studied, 35 percent were report generation and inquiry/simple analysis, and 50 percent were complex analysis. Fifty-two percent of the systems were departmental in scope and 31 percent were personal. Thirty-six percent of the data used for user-developed applications was extracted from production files. In the firms studied, most of these systems were developed by functional support personnel for non-programming end users.

Of the seven firms Rockart studied, only two had an information center. None of the firms had a strategic plan, control policies, or a means for assessing application development priorities for end-user computing.

The question of whether information centers have enabled firms to reduce their application development backlog has been explored in several studies. In a report on end-user computing in 71 medium to large corporations, plus two U.S. government agencies, 33 percent of the organizations reported a backlog reduction that at least came up to their expectations; 7 percent said that any reduction was less than expected; and 60 percent said that it was too soon to tell the effect of the backlog. (9)

A closely related question was whether the use of fourth generation tools in application development would reduce development time. In fact, 77 percent of the companies observed a reduction in development time, and about half of these firms reported that end users were using the tools directly.

In his study of end users as application developers, McLean identifies three types of computer applications: personal applications, which are designed to serve the needs of the individual; departmental applications, which provide the reports, queries, and analyses for the department's information system; and corporate applications, which involve data from several departments. (10) Personal applications, which include queries and ad hoc reports, can reduce the workload of data processing professionals. Departmental applications which use local data files often require data processing assistance in requirements analysis and technical support. Corporate applications follow the traditional life cycle, but fourth generation tools can be used in prototyping, establishing requirements specifications, and developing documentation.

In his study of the impact of user-developed applications on the backlog, Robert Rosenberger argues that end-user computing consists mainly of the ill-defined "one-shot" types of personal and departmental applications which do not significantly impact the DP backlog. The backlog of larger production systems projects is not necessarily lessened. (11)

Users, according to McLean, are in an excellent position to impact the maintenance problem, particularly those changes designed to extend or enhance the features of a system. If users are equipped with the tools to make simple queries, generate reports, and undertake analyses, most of this maintenance can be offloaded from DP professionals. (12)

In a study by Rivard and Huff, available literature was used to define the success of user-developed applications as resulting from a decrease in the DP application project backlog and a decrease in the DP maintenance load. However, interviews with DP professionals in ten organizations revealed that the primary aspects of success were user satisfaction with DP services, improved user productivity, decreased use of outside timesharing, and the assurance that the users were able to use computer resources in a manner profitable to the firm. The authors suggest that users themselves should be held responsible for evaluating the success and cost-

effectiveness of their own applications. (13)

In a survey of end-user computing with 21 firms, Tor Guimaraes identified some of the issues raised by MIS attitudes toward end-user computing. Of the firms studied, seven had personal computers for users but no information center, six had personal computers and no information centers, and eight had information centers but no personal computers for users.

One area of the study was policies affecting user-developed applications. Only two of the companies surveyed planned to establish quality assurance mechanisms for user applications even though most of the MIS executives interviewed expected user applications programmed in procedural languages to be technically deficient in comparison to systems developed by MIS personnel. However, most of the MIS executives had relatively little concern about user incompetence in developing applications. (14)

The main concern expressed by the MIS executives was the potential problem of mainframe data contamination. To guard against this, users were not able to upload data to production data files. However, they could download extracts of production data to do their queries and reports.

To overcome the problems of user-developed applications, Guimaraes argues that the internal audit function should be responsible for their quality and that MIS should establish systems development standards. Such standards would facilitate systems development, maintenance, sharing of systems, and training. Rockart, too, recommends that policies be designed to address the documentation, data security, and data management requirements of user applications. (15)

A number of issues are raised by recent studies of end user applications. Most user-developed applications are queries or reports from extracts of production data files. As such, most of what users are doing is not likely to affect the new application development backlog. However, the prototyping of applications using fourth generation tools can improve the productivity of traditional projects by providing better requirements specifications and design documentation.

Information center analysts should help define application requirements for user-developed systems. This means determining if a particular application can best be supported by using microcomputer-based software, an office automation system or a mainframe-based application generator.

Requirements for documentation of user-developed applications should be established and include data file definition, input/output specifications, logic, controls, and operational procedures. Applications which are transportable to other departments should be identified to prevent the tendency to re-invent the wheel.

Controls over hardware and software will be necessary to assure that microcomputers are compatible with the corporate data processing network, have access to host-based data files, and can share network resources such as high-speed electronic printers and hard disk storage devices. Control over application development will require user managers themselves to identify priorities for end-user computing.

The growth of end-user computing entails a process of organizational learning which Chet Mills describes. In the first stage, users satisfy their individual data needs by making queries and generating reports. After initial success, applications requiring more complex logic are developed during a second phase. In a third phase, users recognize that multiple applications share the same data, and efforts are made to consolidate data, minimize redundancy, and improve data integrity.

In the fourth phase Mills describes, users begin to extend existing applications and in some cases require very sophisticated application software. Now, the information center evolves into an application development center, using more traditional systems development techniques and technologies. In the last and fifth phase, information center analysts are moved into functional groups to work with users to develop information systems that support business plans. (16)

The phases which are defined by Mills depict a stage evolution which not only involves the growth of technology but also a process of organizational learning. To support this learning curve, information center analysts will need to provide the training and support which will be critical in moving from one phase to the next.

References

- (1) James Martin, Application Development Without Programmers, (Englewood Cliffs, NJ: Prentice-Hall, Inc., 1982), p. 55.
- (2) Ibid, p. 83, 84.
- (3) Chester Mills, "The Information Center," DRS Journal, V. 1, No. 1, 1983, p. 4.
- (4) Ibid, p. 7.
- (5) "Future Effects of End User Computing," EDP Analyzer, November 1983, V. 21, No. 11, p. 4.
- (6) Ibid, p. 3.
- (7) "Coping with End User Computing," EDP Analyzer, February 1984, V. 22, No. 2.
- (8) John Rockart and Lauren Flannery, "The Management of End User Computing," Communications of the ACM, October 1983, V. 26, No. 10, pp 779, 780.
- (9) "User Driven Technologies: Personal Computers, Information Centers, and Fourth Generation Languages," (Port Jefferson Station, NY: FTF Technical Library, 1983). Reported in EDP Analyzer, February 1984.
- (10) E.R. McLean, "End Users as Application Developers," MIS Quarterly, V. 3, No. 3, December 1979, pp 42, 43.
- (11) Robert Rosenberger, "The Productivity Impact of an Information Center on Application Development," Proceedings GUIDE, No. 53, Dallas, Texas, November 1981, pp 918-932.
- (12) McLean, p. 44.
- (13) Suzanne Rivard and Sid Huff, "User Developed Applications: Evaluation of Success from the DP Department Perspective," MIS Quarterly, March 1984.
- (14) Tor Guimaraes, "The Benefits and Problems of User Computing," Journal of Information Systems Management, V. 1, No. 4, Fall 1984, pp. 8, 9.
- (15) Rockart, p. 784.
- (16) Mills, pp. 42-46.

THE PSYCHOLOGY OF EFFECTIVE PROMPTING

by

Dr. Gerald N. Pitts

Trinity University
San Antonio, TX

ABSTRACT

The design of the interface between the system and the user must consider the psychology of the user as well as the hardware and software constraints of the system. Many users are often disappointed to find that their productivity has been significantly reduced by factors such as cumbersome data entry procedures, obscure error messages, inconsistent procedures, unclear error handling, and/or confusing sequences of conflicting directions. Since computers were developed "by the people", does it not stand to reason that computers should be designed "for the people" as well? The need for a holistic systems design resulting in a highly adaptive, flexible, interactive graphical man-machine interface is crucial.

Introduction

I have chosen the field of associative language theory to explain the predictive quality of natural language. Of all existing language theories, it alone accounts for predictability of language, as it examines word frequency, latency (comprehension ratio), and classification structures.

The appropriateness of verbal association in regard to prompting cues is essential in this discussion, for if one can objectively and extrinsically measure the way in which the vast majority of people use language, you have not only the groundwork for asking the right questions (to get desirable answers), you have an information theory as well. For these reasons I have examined verbal association and propose to explore the following guidelines for optimal prompting systems:

- (1) Word frequency
- (2) Latency or comprehension
- (3) Types of association (synonyms, contrasts, antonyms, subordinates, superordinates, coordinates, completions)
- (4) Two Associative Laws
- (5) Proper Sentence Structure (not discussed in this paper)

After weeks of exhaustive and intensive research, I have discovered that the field of prompting dialogue in the computer interface system is one that has been seriously neglected if not virtually ignored by the technicians and programmers, as perhaps they feel that it is the job of the linguist or professional language theorist to develop such a system. Prompting as a discipline or systematic study is a relatively unexplored area, yet the need for natural language intercourse between computer and human is universally recognized by the computer industry and society at large. The key in successfully designing and implementing a highly interactive interface system lies not in formal computer language applications, but in the study of linguistics explicitly, as it applies to natural language, in which highly associative aspects of verbal behavior and theories of meaning are primary concerns.

A fundamental goal essential to the development of effective prompting systems lies in the acquisition of a tool which entails a word-recognition system. Such a system has a highly associative quality, and it has been found that verbal association is an explanatory approach to many aspects of language behavior that implicate meaning, when interpreted in modern terms. The associative definitions and theories place their emphasis on the pattern of verbal associations that a word elicits. These approaches represent the combined influence of early word-association research, and the role-learning tradition that began with Ebbinghaus and Watson's behaviorism.

The experimental study of word associations began with Sir Francis Galton who was concerned with the association of ideas. He looked at 75 words individually, starting a stopwatch the moment he saw each word and stopping it as soon as two different associated ideas had occurred. He observed that different types of words give rise to different types of associations: some elicited purely verbal associations most frequently, and others were as likely to arouse visual images as verbal associations. Later research has emphasized especially the nature of verbal associations. Galton's test of association of ideas became a word-association test. Since one need only analyze the discrete words produced by the subject (i.e. verbal stimuli would produce verbal responses which could quantifiably be evaluated) the stimulus-response theory of the behaviorists investigators could be directly applied to verbal stimuli and verbal responses. In the earliest studies, subjects were instructed to respond as quickly as possible to each word with another word. This discrete free association was discovered to have three basic attributes, namely commonality, latency and classificatory types.

Commonality is the frequency with which an association is given to a work by a large group of people. The discovery of discrete-free association was of immense importance, as it illustrated the universality of some components of language. For example, in studies done by Kent and Rosanoff, discrete free associations for approximately 250 words were given by 1000 subjects. They found that certain words are given by many subjects as responses, but others are given rarely. Another attribute of the frequency is that it often has a characteristic shape, i.e. a few associations are produced by a few subjects. For example, responses to the stimulus word needle in the Kent-Rosanoff norms are as follows:

thread	160
pin	158
sharp	152
sew(s)	135
sewing	107

Note a rapid drop in frequency:

steel	53
point	40
instrument	26

The remainders were 1 subject answers per/each, for example, blood, broken, camel, and crocheting.

The most frequent responses like thread have high commonality, unlike blood which has only 1 occurrence in one thousand. The most frequent response in this example, i.e., thread is commonly referred to as the primary response.

The second aspect of association is the relation between the frequency of an association and its latency or reaction time. Tests have revealed that frequent associations occur more quickly than less frequent associations. The findings of studies conducted in this area reveal that infrequent associative responses have greater conditional uncertainty than frequent ones, and increased uncertainty results in longer reaction time.

Finally, the third feature of associations is their qualitative quality. The distinctions between kinds of associations is psychologically profound. The more common classificatory systems have distinguished such types of association as synonyms (swift-fast), contrasts or antonyms (fast-slow), subordinates (animal-cat), superordinates (hammer-tool), coordinates (dog-cat), completions (sharp-needle), assonant or rhyming associations (pin-tin), and various personal or egocentric associations (lonesome-never).

Some investigators (notably Woodworth) have grouped the various associative types into a smaller number of psychologically useful categories. Woodworth proposed four general categories:

- (1) definitions
- (2) completion and predication
- (3) coordinates and contrasts
- (4) valuations and personal associations

Another very valuable category introduced by these studies include the distinction between *syntagmatic* and *paradigmatic* associations which appear in psycholinguistic literature. Syntagmatic associations are ones that could follow each other in a sentence (table-chair, run-walk, in-out), hence this category corresponds to completion and predication in Woodworth's scheme. Paradigmatic associations are from the same grammatical class, hence they could replace each other in a sentence (table-chair, run-walk, in-out). This category subsumes most of the associations classified under coordinates, subordinates, superordinates, synonyms, and antonyms in classification schemes.

There is some evidence that the frequency distribution of associations conforms to Zipf's Law. Zipf was concerned with the frequency of occurrence of different words in ordinary language use, rather than with associative frequencies. He related the frequency of different words to the rank order of their frequency by charting out the frequency of words by ranking them on a chart and comparing different words from American newspapers plotted against the rank of words when ordered with respect to the frequency of occurrence (both on a logarithmic scale). Rank 1 is the most frequent word, Rank 2 the second most frequent, and so on. The ordinate shows the frequency with which each word occurred in the text (newspaper). The results of this study were astounding. Not only did the two sets of scores correlate to a high degree, but when plotted on logarithmic scale, the result was a straight line! Expressed in a different way, the product of frequency and rank is an empirical constant ($fr=C$), hence the term Zipf's Law. Perhaps, not surprisingly, the same logarithmic relation holds between the frequency of an associative response and its rank order of frequency. Thus ordinary language usage and word association responses appear to obey very similar laws. This is extremely important because it is evident that the word-association test reveals one very important, dynamic component of natural language behavior. Further studies were conducted (Howes) that determined verbal behavior in the association experiment is statistically equivalent to "natural" verbal behavior. There are numerous examples in which association norms are predictive of verbal behavior in a variety of situations.

The associative structure and grammatical structure is an area which was explored as a result of the discovery of verbal associative behavior. These examine the relations among associative distributions, associative meaning, and grammatical class. It was revealed that grammatical classes have larger and more diffuse structures than do semantic associations. No sharp line can be drawn between grammar and meaning, but meaning is definitely influenced by relations in the natural world, which fall outside the scope of the verbal associative approach.

Associative relations among words of various form classes were examined in this study, particularly the paradigmatic-syntagmatic distinction. It was found that form classes differ systematically in the degree to which they yield associations of the two types. The associations to nouns are primarily paradigmatic, i.e., nouns predominantly yield other nouns as associates. (For example, movement yields motion.) On the other hand, adverbs are overwhelmingly syntagmatic in their associations, i.e., the associates are words that occupy different positions within phrases or sentences. (For example, the adverb amazingly yields right, new, accurate.) Verbs and adjectives fall in between, yielding about the same proportions of paradigmatic and syntagmatic associations.

The associative structures of nouns and adjectives led to the two basic observations about association:

- (1) The fundamental relational scheme for nouns is *grouping*. Some groupings are very well-defined, such as dog-cat, but other groupings do not have such simple names (bees-flowers).
- (2) Familiar adjectives often yield *contrast* or *antonyms* as associates, for example, alone-together, active-passive, alive-dead, back-front, etc. (Such contrast pairs accounted for 29% of the 278 common adjectives sampled in these studies.)

The structures of these laws are developed partly through experience with perceptual objects and social usage which requires verbal elements, and new relations constantly arise as the social and non-social context changes. This observation is of the utmost importance because it clearly indicates that associative meaning is not fixed, but rather a dynamic component of language. The underlying associative reactions are allowed to vary as the situational context changes.

The value of the verbal association approach cannot be underestimated, especially in regard to the predictive qualities of language. This becomes of ultimate value in the discussion of prompting systems, for if one can anticipate the way in which most people use a language, then it stands to reason that one can clearly develop the correct criteria for asking the right question. Verbal association is an objective and reliable approach to the study of meaning. It can be used as a measure of meaningfulness. The number of different associations to a given word provides an estimate of sequential dependency between the word and ones that might follow, therefore relating the association technique to information theory. The associative clusters derived from factor analysis of associative overlap data reveal very important semantic characteristics of linguistic concepts, such as synonymy, antonymy, and membership in common conceptual categories, as well as suggesting basic ways in which information can be structured in long-term memory, as for example, with contrast and grouping. The positive features of the verbal associative approach to meaning is of immense significance and consequence in the understanding of language and in the discussion of the development of effective prompting in the interface system.

OVERVIEW OF THE FOURTH GENERATION LANGUAGES MOVEMENT
WITH ACADEMIC USE INTEGRATED

by

Boulton B. Miller
and
John F. Schrage

Southern Illinois University at Edwardsville
Management Information Systems Department
Campus Box 106, Building II
Edwardsville, IL 62026-1001
(618)-692-2504

ABSTRACT

The term "fourth generation languages" (4GLs) is foreign to most computer programs but will become as necessary as COBOL within the next few years. The programming backlog has spurred the information centers concept with the 4GLs as one tool for the user. The 4GLs area is examined and defined with an overview of existing packages.

APPLICATION DEVELOPMENT BACKLOGS

Organizations throughout the country are suffering with the common problem of backlogs in the development of computer applications. At the 1982 Association for Computing Machinery (ACM) meeting in Dallas, statistics were noted that at least thirty month backlogs existed. This did not include the invisible backlog identified by James Martin as those applications users do not submit because they realize their requests would be added to the queue (Martin, 1982).

Additionally, one of the primary reasons why organizations have so readily accepted personal computers has been the application development which can be accomplished on the devices. Even so, many did not realize the increasing use of personal computers until the Portia Isaacson panel presentation at the National Computer Conference in 1983 (Issaacson, 1983). Even her forecast was exceeded by International Data Corporation's statement that the "U.S. personal computer market will surpass mainframes by 1984" (International Data Corporation, 1984).

INFORMATION CENTERS

The information center concept came into being about a decade ago at IBM Canada (Harmond, 1982). The concept originated with the objective to help users learn to help themselves by generating many of their own reports and in developing some of their own computer applications. The information center concept is also needed to improve the typical systems methods by encouraging the use of automated design, fourth generation languages, and prototyping (Martin, 1983). The information center should support a natural division of labor between the end users and the data processing staff (Martin, 1982). However, the organization's information manager must provide the necessary policies and procedures for the control of the organization's data wherever they are used.

When the concept was first applied in our academic program in 1982, it was impossible to locate information centers. IBM came to the rescue and invited students to attend their information center training sessions along with regular customers. Now, within the St. Louis area alone, there are forty or more information centers available for students to evaluate. With one of the authors as a member of SLICE, the St. Louis Information Center Exchange, it has been possible to visit many of the centers and

become acquainted with their managers. The organization of the many information centers has not eliminated the applications development backlogs. As users become more familiar with what computers can do, they request increasing amounts of computer support. In the long run, the information center staffs will be accomplishing what should have been done for many years -- help users to become more involved in the use of computers to obtain the required information for use in decision making. These information centers can provide the means toward the education and control of users as they acquire the estimated 20 million microcomputers in the U.S. business world alone by 1987 (International Data Corporation, 1984). By that time users will be doing half or more of their own computing.

BACKGROUND OF 4GL DEVELOPMENT

Nomenclature is a continuing problem in the information industry. The terminology for fourth generation languages (4GLs) is an excellent example of nomenclature development. Another term often used with similar meaning is "application generator". Vendors often use both terms to describe their products, depending upon the preferences of the prospective buyer. For this material, an assumption is the acceptance of machine language as the first generation and assembler as the second. Third generation languages have included the term procedural with the examples of BASIC, COBOL, FORTRAN, PASCAL, PL/1, and ADA along with many others.

The term "nonprocedural" has been used to describe those programming languages that permit the user to specify what is to be accomplished rather than the detailed how used in the more common procedural languages (Kull, 1983). An application generator where forms are filled in by the user is said to be nonprocedural, but most professionals say that using such a generator is not programming. Most fourth generation languages are described as nonprocedural; however, some resemble third generation procedural languages more than others, and some have both qualities (Martin and McClure, 1983).

FOURTH GENERATION LANGUAGE DEFINED

The term fourth generation language was originated by Nigel S. Read and Douglas L. Harmon in a 1981 Datamation article. According to James Martin "for a language to be worth calling 'fourth generation,' the following characteristics should be present (Martin, 1982):

- (1) User-friendly software.
 - (2) Useable by nonprofessional programmer.
 - (3) Directly employs a data base system.
 - (4) Code requires an order-of-magnitude less instructions than COBOL.
 - (5) Nonprocedural code used where possible.
 - (6) Makes intelligent default assumptions about user wants.
 - (7) Designed for on-line operation.
 - (8) Enforces or encourages structured code.
 - (9) Easy to understand and maintain another person's code.
- (A) Non-DP users can learn subset of language in a short time.
 - (B) Designed for easy debugging.

4GLs FROM MAINFRAME DATA BASE MANAGEMENT SYSTEMS

For a long time one of the authors was convinced that application development tools were only usable by computer professionals. APL is not a representative 4GL; however, it was a pioneer tool for use in applications development -- by professionals, not functional area users in management, marketing, or the personnel department.

A number of languages recognized today as 4GLs originated in connection with mainframe data base management systems. NOMAD, now known as NOMAD 2, was available through National CSS for many years and is now a product of Dun & Bradstreet. MAPPER earned an excellent track record and continues to be provided by Sperry. RAMIS, now RAMIS II, is a well known product of the Mathematica Products Group. FOCUS was designed by Gerald Cohen who was instrumental in the early development of RAMIS, then departed to form Information Builders. Those individuals who use ADABAS are familiar with NATURAL from Software AG of North America, Incorporated. Others, who are familiar with TOTAL from Cincom Systems are probably using MANTIS. Our university uses Cullinet's IDMS and are considering the enhanced IDMS/R with its 4GL. These examples are representative of 4GLs originating in mainframe data base management systems; but these references are not all-inclusive. However, most of these 4GL examples are becoming available for use on microcomputers in either a stand alone mode of operation or as a subset on the PC as a mainframe connection.

4GLs ORIGINATING AS FOURTH GENERATION LANGUAGES

In 1981, one author became acquainted with the developers of PRO-IV, a 4GL developed from the beginning to become a user's application development tool as well as useful by professionals (Miller, 1982). This group at Data Technical Analysts (DTA) recognized that the existing 4GL language software packages were too large to be used on microcomputers available at that time. It was during late 1976, that Mr. Sushil Garg, DTA's lead programmer, began to experiment with a concept, now called 4GL, that marked the inception of PRO-IV. Mr. Garg took note of the vast number of common elements in the many different applications he was developing. He began storing these common elements or modules, pre-coded in the computer, then developed an algorithm to link them together to generate the application based on the end user's specifications rather than the conventional coding of a computer program. PRO-IV is designed to run equally well on PCs and larger systems as all of its sophistication resides in approximately 36K. The 1982 National Computer Conference at Houston exhibited PRO-IV demonstrated at the CIE Systems booth. PRO-IV was also demonstrated as ALL on Microdata equipment. During NCC'83, PRO-IV was announced as available on Digital Equipment Corporation's hardware and IBM PC/XT.

SALVO is the first 4GL based on artificial intelligence and relational data base management concepts for the IBM PC/XT. SALVO is designed for a wide variety of personal computers requiring only 64KB. The originators of SALVO recognized that many so-

called relational data base management systems designed for personal computers were not truly relational systems. They used the design criteria that resulted in agreement with E. F. Codd's definition of a true relational system with the following components (Codd, 1982):

- (1) Relational data structure, i.e., flat files, relations, tables;
- (2) collection of relational algebraic functions or rules of inference; and
- (3) collection of relational integrity rules.

The addition of artificial intelligence functions to SALVO represents a departure from many fourth generation products written for mainframe computers.

AS INTEGRATED SOFTWARE FOR MICROCOMPUTERS

Recent articles and advertisements appeared indicating fourth generation status for a number of software packages for microcomputers. One example was a listing in Technology Transfer News, updated in the following issue to be more specific as to whether or not fourth generation should be used, or in some cases where other products were required to give the items a 4GL status. The update made the earlier evaluation similar to a description of twenty-four relational data base systems appearing in the same timeframe (Bowerman).

A number of software packages advertised as integrated packages probably will also attain recognition as 4GLs. Software vendors continue to sell separate packages for specific applications such as word processing, business graphics, data base management, numerical analysis, decision support systems, report generators, data dictionaries, online and batch applications development tools, communications, and others. These products have continued as separate items because the market has continued. Users are now demanding integrated software. A good example is Applied Data Research ADR/IDEAL offered as an advanced application development tool for end users, programmers, and systems analysts. ADR/IDEAL is an excellent example of an integrated concept although not originated as micro-based software but being designed for their use.

More specific integrated software for microcomputers with yet unproven 4GL track records are DESQ from Quarterdeck Software, VISION from VISICORP, MS-WINDOWS from Microsoft Corporation, and SYMPHONY for use with LOTUS 1-2-3. Other products to watch during development and maturity are MILLENIUM from the McCormack & Dodge Corporation for use on IBM's Personal Computer XT/370; APPCEN, and application generator from the Software Express; and UMBRELLA from Hogan Systems. Additional views on the development of fourth generation tools appear in a special report of Computerworld demonstrating the timeliness of this subject (May 23, 1984).

PC/FOCUS

When using Martin's book Application Development Without Programmers, a major point is that much programming had to be done to make the new tools work. For example, the PC/FOCUS software requires about three million bytes and is delivered on eleven double-sided/double-density floppy disks. PC/FOCUS has been used since June 1983 in our MIS program, witnessing the gradual improvements as new releases have been received. Distributed Processing with FOCUS' micromainframe link allows PC/FOCUS users to process transactions in full-screen (FIDEL) mode, against a data base on the mainframe computer. Multiple users will be able to access and/or update the same central data base simultaneously. The new release of PC/FOCUS has add several needed areas: TABLETALK Visual Report Generator; FILETALK allows the user to interactively create Master File Descriptions; ANALYSE Statistics; Visicalc and Lotus 1-2-3 Interface; and Bisynchronous 3270 Interface.

The equipment presently having the package is an IBM PC/XT which uses 256K of primary storage plus a circuit board with an additional 512K of RAM. This accelerator board was acquired when PC/FOCUS relied on its circuitry for access control. Later releases dropped the necessity for the circuit board as a control device and added a floppy disk for use to move from PC/DOS to PC/FOCUS. The XT uses a standard IBM 10 Mb hard disk, unfortunately, without a streaming device.

END USER COMPUTING

Of major concern today is the fear that top managers throughout all organizations will not realize the spread of end user computing. The previous reference to the 20 million microcomputers in American businesses by 1987 is alarming. It is not the number alone that gives the most concern, but the realization that users have little mainframes on their desks. The microcomputers are only the beginning; as work stations designed around 32-bit microcomputers with built-in telephones provide capabilities users are not educated to handle. Neither do most top managers realize the added problems that will accompany these new devices with their improved software. Users can organize their own functional area data bases, generate their own reports, answer their own what-if questions, apply business graphics, spread sheets, statistical applications, and decision support systems without any help from the central computer center and the computer professionals.

THE CLASSROOM

The 4GLs are really not the future but the present. While the components might not be available for actual use in the classroom, the theory on the topic must be included for both the information systems major and business administration student. After mainly providing only lecture for over two years, students have hands-on applications using the PC/FOCUS and MAPPER packages in the class. Additionally, some area organizations have added the 4GLs to their production programming mode with the information center. At least four vendors have provided instructional materials and demonstrations for students. At least three former students now are working in organizations with the 4GLs are their major tool.

Future 4GLs' classes will increase the practical applications using at least two packages and students finishing the first course are asking for more such that they can more critically evaluate the material for use in their companies. The Chief Academic Officer of the University believes that industry leaders need more exposure to that aspect of information systems by providing a summer grant on information systems tools to a post-Master's group of industry users.

During the past decade, progress has been made in the use of data base management software and centralized control of important organizational data. Reports from computer and data processing centers can be trusted because the data edited, inputs and updates are controlled, and responsibilities established. Of course, there are many things we would like to do better, such as metadatabases, information resource management, and decision support systems, but progress has been rewarding. On the other hand, users might revolt and attempt to retain their own functional area data, process the data on their own workstations without sufficient supervision to retain the organization-wide data/information management needed to meet the needs of the organization as a whole.

4GLs AND THE FUTURE

It may be too early to clearly define fourth generation languages. However, their acceptance to provide users with the capability of acquiring their own reports and developing some of their less sophisticated applications is impossible to deny. Similar uses by computer professionals to speed up programming and application development are being accepted. As prototyping techniques are perfected for use during the development of more sophisticated applications, 4GLs will become more popular than they are today (Alavi; Boar; Canning). A major area for development appears to be 4GLs used with automated structured system design tools.

An integrated tool reported to improve user's development of applications on an IBM PC/XT is EXCELERATOR from Index Technology Corporation. This software product was developed to assist in all phases of systems analysis, design, and documentation. User acceptance is not yet available (Lambert). Another development tool designed to run on UNIX V and reported to be in use on about twenty different items of hardware by 2,000 users is APPGEN from the Houston company Software Express (Thomas, 1984). These examples demonstrate that there are a number of vendors attempting to develop and market similar products.

REFERENCES

Alvai, Maryam. "An Assessment of the Prototyping Approach to Information Systems Development," Communications of the ACM, 27:563, June 1984.

Boar, Bernard H. Application Prototyping. New York: John Wiley and Sons, 1983.

Bowerman, Robert. "Relational Database Systems for Micros." Datamation, July 1983, 128-34.

Brown, Marc, Norman Meyrowitz, and Andries van Dam. "Personal Computer Networks and Graphical Animation: Rationale and Practice for Education," AEDS Monitor, May/June 1983, 15-24.

Codd, E. F. "Relational Database: A Practical Foundation for Productivity." Communications of the ACM, 25:111, February 1982.

Canning, Richard G. ed. "Developing Systems by Prototyping," EDP Analyzer, 19, September 1981.

EDP Industry Report, The International Data Corporation, July 8, 1983.

Hammond L. W. "Management Considerations for an Information Center," IBM Systems Journal, 21:145, 1982.

Higher Order Software Seminar, "Automation of the System Development Process," Milwaukee, WI, 21 June 1984.

International Data Corporation Executive (IDC) Spring Executive Conference, Phoenix, AZ, 29 April-2 May, 1984.

Isaacson, Portia, and Benjamin M. Rosen. "Personal Computing Industry: The Experts Forecast the Future," Paper presented at National Computer Conference 1983, May 16, 1983, Anaheim, California.

Konsynski, Benn R., and Jay F. Nunamaker. "PLEXSYS: A System Development System," Advanced System Development/Feasibility Techniques, May 1982.

Kull, David. "Nonprocedural Languages Bringing Up the Fourth Generation," Computer Decisions, December 1983, 154-62.

Lambert, Jeanne T. Index Technology Corporation, Cambridge, MA. Letter to the Author, 1 June 1984.

Martin, James. Applications Development Without Programmers, Englewood Cliffs, NJ: Prentice-Hall, 1982.

Martin, James and Carma McClure. Software Maintenance -- The Problem and Its Solution, 1983.

Martin, James. An Information Systems Manifesto, Englewood Cliffs, NJ: Prentice-Hall, 1984.

Martin, James. James Martin Seminar on Information Systems Manifesto, Chicago, IL, 25-29 April 1983.

McDonnell Douglas Corporation. Press Release 80-104, 13 August 1980.

Miller, Boulton B. Computers and Data Processing, Edwardsville, IL: Bainbridge, 1982.

Rauch-Hindin, Wendy B. "Mainframes and Micros," Systems and Software, June 1983, 68-89.

Read, Nigel S. and Douglas L. Harman. "Assuring MIS Success," Datamation, February 1981, 109.

Technology Transfer News, 1 no. 2 (1983) and 2 no. 1 (1984).

Teichrow, Daniel, and Hasar Sayani. "Automation of Systems Building," Datamation, August 15, 1971, pp. 25-30.

Thomas, Steve. Software Express, Houston, TX. Letter to Dr. Miller, 15 June 1984.

Thomas, Steve. Software Express, Houston, TX. Telephone conversation with Dr. Miller on 29 June 1984.

Uhrbach, Harold. "Trends in Distributed Data Architecture," Presentation at SIGBDP Breakfast Meeting, ACM 1983, October 25, 1983, New York.

Wall Street Journal, Midwest edition, 5 March 1984, 26.

EFFECTIVE END USERS AND FOURTH GENERATION TOOLS

Dr. Carol Chrisman
Dr. Barbara Beccue

Illinois State University
Applied Computer Science Department
Normal, Illinois 61761
(309) 438-8338

ABSTRACT: Companies are experimenting with the use of fourth generation tools in developing application systems. Two new development approaches which incorporate fourth generation tools are Information Centers and prototyping. Each of these approaches rely on extensive and active involvement of the end user. In order for these approaches to be effective, end users must be educated about fourth generation tools. This education should include knowledge about what types of tools are available and when to use them, in addition to experience using specific tools.

INTRODUCTION

Data processing departments have generally not been successful in keeping pace with requests for new applications. This has resulted in large backlogs of application requests; consequently, end users often must wait a period of years for a request to be satisfied. Because this delay is unacceptable, companies are experimenting with alternate approaches to developing application systems.

Companies are trying to use new development tools, fourth generation tools, which will allow application requests to be satisfied more quickly. Since these tools automate portions of the application development process, they make the data processing staff more productive. While these tools are very powerful, their use does not always require extensive data processing knowledge. Therefore, it is feasible for end users to partially satisfy their computing needs through the use of fourth generation tools. If end users can become more self-sufficient, they can bypass the application backlog.

As companies have experimented with the use of these new fourth generation tools, two new development approaches have emerged. These two new approaches are Information Centers and prototyping. Each approach incorporates fourth generation tools and relies on extensive and active involvement of the end users. Since end users play a major role in these approaches, they must be educated about fourth generation tools. In order for the end users to be effective, they need to know about the available types of tools and when and how to use them.

INFORMATION CENTERS

One of the new system development approaches which many companies are utilizing is to have the end user do their own application development. This approach is frequently implemented through an Information Center [5,6,7,8] which is an internal organizational unit whose purpose is to aid the end users in satisfying their informational needs. The Information Center approach is beneficial because of its responsiveness and flexibility. In order to make knowledgeable decisions, end users often need information more quickly than it can be provided by traditional data processing services. When end users function within a dynamic business environment, the information needed for future decisions often cannot be specified completely at the time a system is

developed. In this approach, the fourth generation tools provide the flexibility for end users to obtain needed information on a timely basis.

The Information Center provides a setting in which end users can satisfy some of their own computing needs. The Information Center provides fourth generation tools which are suitable for end users and a data processing support group which will train and assist the users. In this setting, the user can access corporate data to generate reports, to answer "what if" questions, to obtain statistical results, and to select and examine data in different ways. In some cases, the user will also set up and maintain specialized local data through Information Center facilities.

The uniqueness of the Information Center approach lies in the fact that the end user does the development work. The end user determines the requirements and then designs and implements the reports or simple application himself. The role of the support group in this process is to act as consultants. Initially, there is a lot of interaction between the end user and the support group but dependence on the support group decreases as the end user becomes more experienced with the Information Center facilities.

PROTOTYPING

A second systems development approach which relies heavily on end users and fourth generation tools is prototyping [2,3,4,6,7]. With this approach, a prototype or working model of the system is created quickly using fourth generation tools. In some instances, the final prototype model is used as the production system. In other cases, the final model is used as a statement of the system requirements and a production system is implemented using a procedural language such as COBOL.

The prototype is not built by the end users themselves, but rather through a cooperative effort involving the end users and a data processing professional. The expertise in the use of the particular fourth generation tools is provided by the data processing professional. The role of the end users is to provide information on the system requirements and to evaluate the prototype and help in its refinement. Since the prototype is a working model, part of the evaluation consists of the users trying out the model to determine its acceptability. The evaluation and refinement of the prototype is repeated until an acceptable model is created.

APPROPRIATE FOURTH GENERATION TOOLS

In recent years there has been a proliferation of tools which automate portions of the application development process. Many companies are incorporating these new fourth generation tools into their system development approaches. Some examples of fourth generation tools are: FOCUS, NOMAD, RAMIS, INFO, MAPPER, and APL [1]. One characteristic of most fourth generation tools is that they are nonprocedural as opposed to third generation procedural languages like COBOL.

Fourth generation tools are often classified according to the intended purpose of the tool. Common classifications are: report generators, query languages, application generators, and data management or database systems. Some tools are very powerful, serving multiple purposes through an integrated approach. Examples of multi-purpose tools include most database systems and products like FOCUS which is not dependent on any particular database system.

Many tools are appropriate for use in both Information Centers and prototyping. The nonprocedural nature of most of the fourth generation tools which are used in prototyping make them easy enough for end users to use effectively in Information Centers. Some tools tend to be found in only one of the development environments. For example, special purpose packages such as financial planning would be more apt to be used only in an Information Center. Fourth generation tools that need more data processing knowledge would more likely be used in prototyping.

EDUCATION

The end users play a major role in making the development approaches of Information Centers and prototyping work. In order to derive maximum benefit from these approaches, end users should be educated about fourth generation tools. In prototyping, the end users need only basic knowledge about the fourth generation tools involved, since the working model is built by a data processing professional. In the Information Center approach, more expertise in fourth generation tools is needed by the end user since he is doing his own development. End users need to know what types of tools are available and the capabilities and typical uses of each type. In addition to knowing about the types of tools, the end users need to know when it is appropriate to use a particular type. In order to become effective, end users also need knowledge about and hands-on experience with specific tools.

Information about the different types of fourth generation tools can be obtained from a variety of sources. For example, readings from professional publications, promotional literature, films and demonstrations can illustrate the capabilities and some typical uses of the tools. The appropriateness of a particular tool in a given situation can be demonstrated by the use of examples and case study discussions. This study can help end users learn the limitations and capabilities of each tool and aid them in the selection of appropriate tools. Knowledge of a particular tool can be acquired through the use of tutorials and manuals which include many useful examples in addition to the syntax rules. Examples are needed to illustrate how particular commands or combinations of commands can be used effectively. Expertise with the tool is then gained through systematic use of the tool.

Companies are going to need to educate end users. One way that universities can help in satisfying this need is to provide appropriate courses. Most future end users are not computer science or information systems majors in college. However, many of them do take an introductory computing course which could include some of the needed fourth generation information. An introductory course could only be

expected to expose the student to the concepts involved in these new approaches. A more advanced course which specializes in fourth generation tools from the end users' viewpoint would be needed to provide more in-depth knowledge and experience. In order to reach current end users, appropriate material could be taught in continuing education courses. By careful structuring of courses, the university can play a vital role in educating end users to be effective in the use of fourth generation tools.

REFERENCES

- [1] Specific products mentioned in this paper are meant as examples only and do not represent an exhaustive list of all applicable products. The use of a product name is not meant as an endorsement of that product.
- [2] Appleton, Daniel, "Data-Driven Prototyping", *Datamation*, vol. 29, no. 11 (November 1983).
- [3] Boar, Bernard, Application Prototyping, John Wiley & Sons, 1984.
- [4] "Developing Systems By Prototyping", *EDP Analyzer*, vol. 19, no. 9 (September 1981).
- [5] "Expanding DP Functions to Support the Information Center Concept", *Computing Newsletter*, vol. 16, no. 8 (April 1983), p. 1.
- [6] Martin, James, Application Development Without Programmers, Prentice Hall, 1982.
- [7] Martin, James, An Information Systems Manifesto, Prentice Hall, 1984.
- [8] "Where Will Applications Be Developed", *EDP Analyzer*, vol. 21, no. 12 (December 1983).

TUTORIAL: EXPERT SYSTEMS

James H. Blaisdell
Computer Information Systems
Humboldt State University
Arcata, CA 95521

The following outline supports a one-hour tutorial on Expert Systems. It assumes no prior knowledge of Expert Systems and is designed to meet the following objectives:

- * the learner should be able to define The Team Expert System
- * the learner should be capable of stating at least two examples of successful Expert Systems
- * the learner should be able to state the components of an ideal problem for an expert System
- * the learner should be able to list three categories of Expert Systems application
- * the learner should be able to describe the components of a simple Expert System
- * the learner should be able to name and categorize at least three expert systems tools
- * the learner should be able to judge the credibility of statements read or heard regarding the state-of-the-art of Expert Systems
- * the learner should be able to state a considered opinion regarding the challenges and threats provided by Expert Systems technology

INTRODUCTION

HACSYMA

Definitions

The DENDRAL family

Artificial Intelligence

The OPS family

Expert

The EMYCIN family

Expert System

DIPMETER ANALYZER

Knowledge

EXPERT

Knowledge Engineering

CADUCEUS

Commercial Systems

History

Proof-of-Concept Systems

Mans quest for knowledge

Mans interest in replecating his essence in machines

GOALS OF AN IDEAL EXPERT SYSTEM

Key events leading to the state of Expert Systems Technology

CATEGORIES OF EXPERT SYSTEMS TASKS

Achievements to date

IDEAL PROBLEMS - DIFFICULT PROBLEMS

KNOWLEDGE REPRESENTATION

Small, Commercial Development Systems

INFERENCE MECHANISMS

Research and Development Systems

Basic Concepts

Higher Level Languages for Expert Systems

Unification

"Assembly Languages" for Expert Systems

Resolution

Other development tools

Forward Chaining

Backward Chaining

THE FUTURE OF EXPERT SYSTEMS

EXPERT SYSTEMS TOOLS

SELECTED REFERENCES

Large, Commercial Development Systems

SELECTED SOURCES

THE MATRIX LIFE CYCLE:
AN INTEGRATIVE MODEL OF THE
ORGANIZATIONAL DYNAMICS OF IMPLEMENTATION

Dr. Dennis P. Heaton
Department of Management and Public Affairs
Maharishi International University
Fairfield, IA 52596
515/472-7414

ABSTRACT

As information technology is disseminated into more and more aspects of organizational life, information systems professionals and users alike need to develop conceptual understanding of the dynamics of change in organizations and skills as managers of change. This paper presents a framework for understanding the implementation of new technology in an organizational setting as a process of social change involving the phases of unfreezing, moving and refreezing. It also outlines a sequence of change agent strategies to facilitate the interaction between IS and users and to promote the smoothness and effectiveness of implementation at each successive stage of the life cycle.

THE MATRIX LIFE CYCLE

The Matrix Life Cycle Cycle shown in Figure 1 is based on a review of the MIS and social science literature on adoption and implementation, on the author's original research on cases of successful and unsuccessful implementation (Heaton, 1984), and on the author's experience as a systems project management consultant. The three columns of the matrix show corresponding steps of the life cycle from the points of view of three different roles: that of the user, the developer, and the change agent or implementation specialist. The three major horizontal divisions of the matrix are the three major phases of the change process - Unfreezing, Moving, and Refreezing (Zand and Sorensen, 1975).

The concept of three phases in the process of social change is based on the ideas of Kurt Lewin. Unfreezing is the essential starting point for change; it psychologically prepares people to search for and anticipate a solution to a recognized and important need. Unless the targets of change feel themselves that change is needed and that the proposed new system will be an answer to their needs, then resistance to change can be expected.

After readiness for change has been created through Unfreezing, Moving is the phase in which attitudinal, behavioral and organizational change takes place. This crucial phase, when the system is actually installed, inevitably involves some amount of confusion, and work overload for users and developers alike. Frustration at this point, due to inadequate technical support or inadequate moral support, can also result in implementation failure.

Finally, use of the new system needs to be stabilized during Refreezing. New routines need to be reinforced, refined, and connected to other procedures and structures in the organizational context. Failure to continue to manage the implementation process through this phase can result in systems being underutilized or abandoned.

The Developer column of the chart shows a sequence of development and support activities which are familiar to information systems and automation specialists and hardware and software vendors. The User column shows a sequence of psychological states and activities through which users must progress in parallel with the activities of developers if implementation is to be optimally successful. The Change Agent column indicates a series of strategies for linking users and developers and managing the organizational dynamics of change. Strategies for each of the phases of change -- Unfreezing, Moving, and Refreezing -- are further discussed in the sections which follow.

The Matrix Life Cycle of Innovation Implementation			
INSTITUTION- ALIZATION	CONNECTING	ENHANCEMENTS	R E F R E E Z I N G
COLLABORATION	NETWORKING USERS	CONFERENCES, MAILINGS	
REFINED USE	EVALUATION	ADVANCED COURSES	
MECHANICAL USE	REINFORCEMENT	TECHNICAL CONSULTING	
PILOT USE	CRISIS MANAGEMENT	TECH SUPPORT TRAINING	M O V I N G
DIRECTIVE AND PLAN	IMPLEMENTATION CONSULTING	INSTALLATION PREPARATION	
ACCEPTANCE	RELATING	SELLING	
TRIAL	SCREENING	DEMONSTRATION	U N F R E E Z I N G
INTEREST	PROMOTING	PROGRAM/ACQUIRE	
AWARENESS	INFORMING	DESIGN/SELECT	
GOAL SETTING	CLARIFYING	SPECIFICATION	
PROBLEM SENSING	ASSESSMENT, ANALYSIS	OPPORTUNITY SENSING	
User Perspective	Change Agent Perspective	Developer Perspective	

Figure 1

STRATEGIES FOR THE UNFREEZING PHASE

During the Unfreezing stage, while the developer is producing or acquiring new technology for the user organization, those whose organizational life will be impacted by the technology must be prepared for it. The abrupt imposition of a new system, without any attention the requirement of unfreezing, was the major cause of implementation failure in one case in the author's research.

Managers of change need to understand that the adoption of new technology by users progresses gradually through the stages of awareness, interest and trial before acceptance (Havelock, 1973). A change agent's role is to facilitate progress through these stages by informing users about the possibility of coming change, promoting the advantages of the new technology, and helping users relate to the new technology in terms of their own objectives. While doing so, the change agent needs to work with the user and the developer to screen out those elements of an innovation which would not be in line with important organizational objectives or which would have unacceptable social impacts (Nichols, 1981).

Even prior to the stage of informing change targets about systems which are in development, a change agent can facilitate the process of analyzing needs and defining goals through which decision-makers initiate systems projects. Too often these needs and goals are not clearly defined. Different parties within both the user organization and the development group may hold different expectations for the system (Ginzberg and Alter, 1978). Technically competent systems analysts often complain that they could build the system the users wanted if only the users knew what they wanted!

The change agent skills which are needed during these early stages may be more those of a consulting psychologist than those of a systems designer. By taking the role of a counsellor, Wolek (1975) has suggested, a technical consultant can help users clarify for themselves what the nature of their problems are and what would be the criteria on which they would accept a solution to those problems. The change agent may also need to be trained to diagnose and facilitate the interpersonal and intergroup "politics" of decision-making in an organizational context. The frustration which system developers experience with users may be due to a blind spot with regards to the behavioral dynamics of unfreezing.

STRATEGIES FOR THE MOVING PHASE

Implementation success depends not only on developing receptivity and commitment during the Unfreezing phase, but also conscientious planning (Ginzberg, 1981) and support for users during the initial stage of actual use of the new system. At this stage, according to the strategic framework of the Matrix Life Cycle, an implementation plan should be established which details:

- the activities of user coordinators, IS staff, and possibly outside vendors in getting the system established in use
- the timeframe for spreading introduction to the system from pilot users to other groups, and the timeframe for incrementally implementing nonessential components
- how management is going to kick off and control the implementation project.

A change agent's strategy should be to avoid a plan which attempts to introduce too much change, too quickly, for too many people. He or she would also make sure that adequate staff have been allocated to give timely support to new users during this critical transition period.

When users are introduced to a new technology, they need training and assistance until they get over the initial learning curve. A one-shot training class and a system manual, while better than no training and documentation, are generally not sufficient to enable users to handle the

problems they will encounter when they attempt to use new technology on their own. Ideally, assistance should be available, as soon as the need arises, either through an on-site consultant, or a hotline. Crisis management means that problems which may arise through user error or through technical malfunctions are addressed immediately. During this Moving phase, most projects need a coordinator who is devoted to supporting the implementation project and organizationally placed to be able to resolve problems as quickly as possible.

STRATEGIES FOR THE REFREEZING PHASE

Whereas the Unfreezing phase sought to destabilize the existing order to prepare the way for the introduction of an innovation, the Refreezing phase aims to restabilize a new order, and integrate the innovation so that it becomes a part of routine practice in the organization.

If the user finds the new system improves his own job, then reinforcement is intrinsic in the use of the system. Not all users, however, may perceive that an information system has immediate reward for them; for instance, they may be required to track data for accountability to managers in another department, at another level of the organization, or for government regulators. In cases where a user does not find the system to be intrinsically rewarding, extrinsic reinforcement from management is essential. This may be through feedback which says -- I see that you are using the new system and I approve; or I see that you are not using the new system and I disapprove. Another very important way in which a manager can reinforce the use of a system is to incorporate use of the system into his or her own management practices. For example, to reinforce the use of a project management system by subordinates a manager can ask to see a specific report rather than accepting status reports not generated by the system. At the same time, if users get the picture that managers don't use the system and don't punish non users, abandonment of the system may become the norm.

After users have gotten started, their use of the system and understanding of its underlying concepts may be refined by consulting, and advanced courses on system use. The need for, or the benefits of, such follow-up training can be identified by evaluating users' proficiencies. Evaluation at this stage can also help determine what modifications would enable the system to more effectively meet organizational objectives.

Contact and collaboration with other users within the same organization or in other user organizations can reinforce and refine use of the system during this phase. By sharing experiences, users can get practical suggestions as well as the psychological support of knowing that others are sailing in the same boat. System developers can promote collaboration among users through conferences and newsletters.

An implementation project is over when the new system is institutionalized. Indicators of institutionalization (Yin, 1981) are the survival of the system through personnel and equipment turnover and new funding cycles. To facilitate implementation, a change agent can deliberately connect the new system to as many other routines in the organization as possible, including job description, performance reviews, policies and procedures, and decision-making practices. It may also be necessary to deliberately discontinue old systems or methods to assure that users rely on the new system.

CONCLUSIONS

The organizational dynamics of implementation should be included in IS education. Both developers and users can benefit by becoming better managers of change. The Matrix Life Cycle outlines a framework of implementation concepts and strategies which integrates the perspectives of users, developers, and specialists in the behavioral aspects of change in organizations.

Bibliography

Ginzberg, M. Key recurrent issues in the MIS implementation process. MIS Quarterly, 5,2, 1981, pp.47-59.

Ginzberg, M. and Alter, S. Managing uncertainty in MIS implementation. Sloan Management Review, Fall, 1978.

Havelock, R. The Change Agent's Guide to Innovation in Education. Englewood Cliffs, N.J.: Educational Technology Publications, 1973.

Heaton, D. The process and outcomes of implementing system development methodologies. Unpublished dissertation, Boston University, 1984.

Nichols, M. A behavioral analysis for planning MIS implementation. MIS Quarterly, March, 1981, pp. 57-65.

Wolek, F. Implementation and the adoption of managerial technology. Interfaces, 5, 3, 1975, pp 38-46.

Yin, R. Life histories of innovations: how new practices become routinized. Public Administration, 41, 1, 1981, pp. 21-28.

Zand, D. and Sorensen, R. Theory of change and effective use of management science. Administrative Science Quarterly, 20, 4, 1975, pp. 532-545.

ESTABLISHING AN EXECUTIVE ADVISORY COMMITTEE: A CASE STUDY
M. Khris McAlister, University of Alabama in Birmingham
M. Blaine McCormick, East Tennessee University

ABSTRACT

This paper describes the rationale, goals, and start-up process for establishing an Executive Advisory Committee for the MIS area within the School of Business. In addition, the results from 18 months of operation are reviewed and caveats discussed.

A thorough review of the Management Department curriculum indicated that the department, and especially the MIS area, could benefit from more interaction with the local business community. In an effort to remedy this condition, an MIS Executive Advisory Committee was established. The following describes a procedure for developing such a group and the early results that, in part, are attributed to the committee.

COMMITTEE OBJECTIVES

The faculty and members of the local business community met to discuss the concept and develop a specific plan for an executive advisory group. While there were some minor reservations voiced regarding the strength of the executive's input into the academic process, the group's role was defined to be strictly advisory, thus maintaining the academic independence. It was, however, recognized that the Business School and the MIS program, in particular, would greatly benefit from the insights, experiences, and the knowledge resident in the business community. To clearly enunciate the role of the executive committee, the following objectives were defined.

- (1) To identify curriculum strengths and weaknesses in the Information Systems area in terms of the required preparation students must have to successfully compete for jobs.
- (2) To serve as an interface between the business faculty and the business information systems practitioners.
- (3) To help identify guest speakers or practicing experts to serve as possible classroom lecturers, seminar leaders, etc.
- (4) To help associate faculty research interests with the practitioner's search for problem solutions.
- (5) To help in the counseling of students interested in information systems by being available to share their ideas, knowledge, and experience with students on both a formal and informal basis.
- (6) To provide input into the School of Business recruiting activities for qualified MIS faculty and students.

COMMITTEE MEMBERSHIP

A number of committee membership options were explored. Given the desire for the committee to be an interested and active body, it was felt that a large number of executives would dilute the vitality of the committee. For this reason, the membership was limited to five executives. To achieve maximum contact and participation by the executives in the community and maintain the participation goal, each executive was asked to participate for staggered two year terms.

The executives asked to participate were top level managers of large corporate information resource areas. The firms represented a broad spectrum of the local business community; vice-presidents of a regional utility, a state wide bank, a national insurance holding company, an international energy company, and the manager of a large computer firm.

COMMITTEE MEETINGS

A significant factor determining the success of this type program is the procedure for carrying on its business. The upper level managers participating in the program were constrained by their corporate obligations. To maximize the input from these active executives, the business is conducted, where possible, on an informal one-on-one basis. Formal meetings of the complete committee are limited to one or two per year. With the executives' schedules, this is about the limit for finding a day and time where all are available. This formal meeting limitation did not in any way impede executive participation.

RESULTS AFTER ONE YEAR OF OPERATIONS

After more than one year with the committee, a number of accomplishments can be noted. First, the Department of Management embarked on a plan to offer a management degree with concentration in MIS, Operations Management, and Human Resources Management. (The success of the MIS Executive Advisory Committee lead to the formation of similar committees in the Operations Management and Human Resources Management areas.) The MIS executives reviewed the feasibility of the MIS proposal from the business community's perspective. Second, the executives reviewed and offered very constructive suggestions regarding the MIS course mix and specific course content. Third, the executives actively supported the curriculum by identifying experts within their own organizations to participate in the MIS classes as guest lecturers. For example, the banking executive discussed his firm's disaster planning procedure with a graduate MIS class, and a manager in charge of office automation planning discussed corporate planning policy. It is interesting to note the latter executive was flown in from his home office two hundred miles away for the classroom activity. In addition, one firm's management consultant has participated in both class activities and in presentations during a formal committee meeting.

In addition to these direct curriculum and classroom activities, there have been a number of developments that can be attributed to the executives' participation on the committee. One firm has offered to take two "co-op" students with the MIS concentration when the program develops to the point of having advanced students. Some of the faculty have begun to participate in training and consulting activities with these participating organizations. Discussions with the executives have also helped identify problems that may prove fruitful for research.

The benefits are not all one way toward the academic side. The committee meetings have become a neutral site for the executives to discuss some of their own problems and concerns with their peers. It appears that a new role for the business school has been discovered. Local managers have been able to discuss problems in a focused session with neutral faculty leadership. One session of this type has been held and was quite successful.

CAVEATS

From the time of its conception, the committee was considered to be an academic support activity, not a fund raising institution. For that reason, a conscious effort has been made to avoid financial need topics with the executives. It was felt that financial participation may develop of its own accord over time but was not a focus of the committee. Second, a posture was maintained with the committee that the academic program has much to gain from the expertise in the business information processing community. In addition, the responsibility for maintaining the momentum of the committee rests with the School of Business. The executives are not, at this point in the genesis of the committee, the initiators.

SUMMARY

The executive advisory committee was established to strengthen the ties between the academic MIS program and the local business community. The committee model noted has helped start this process. The anticipated benefits are beginning to be realized. However, as with most worthwhile ventures, there remains much work to be done in maximizing the executives participation and the benefits from their participation.

STRATEGIC INFORMATION SYSTEM
AS THE OUTPUT OF AN OPEN SYSTEM

Madjid Tavana
La Salle University

The Strategic Component of the overall information systems has often been neglected by the information scientists. The purpose of this paper is to present a more balanced approach to information systems design by showing strategic information system as the input of an open system.

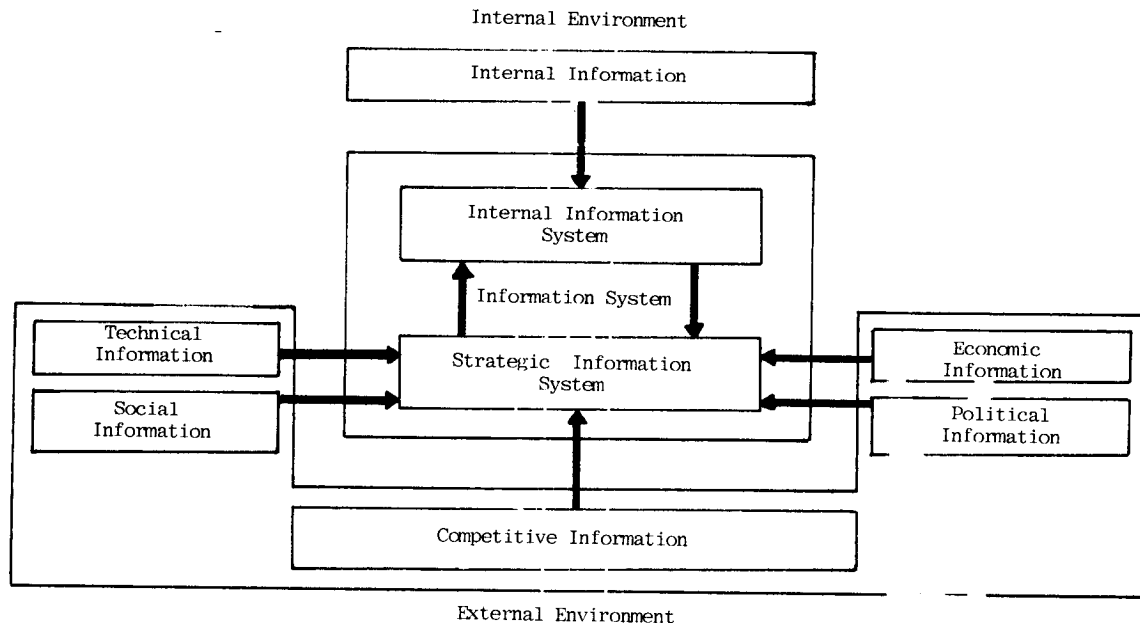
"Open systems import some form of energy" said Kats and Kahn in the Social Psychology of Organizations (1966), in the case of a corporation this energy can be identified not only as raw material, personnel, equipment, money, etc., but also as information that makes it react to different combinations in the conditions of both the external and the internal environments. The subpart of the corporation that facilitates this input is called strategic information system.

The information contained in the internal information system is primarily concerned with routine administration and day-to-day operations within the organization. The internal information system is used for the most part, to support the managerial activities of an organization. There is, however, another category of information essential to the decision making activities. This information consists of reports that support the management activity of strategic planning and facilitates the choice of objectives and the selection of future courses of action.

Strategic information must be seen to be information of a special type. It is information that is necessary to make strategic decisions, information that is capable of distinguishing between the alternative courses of action available to the manager.

To be valuable, the information must have certain characteristics that distinguish it from simple information. It must be correct, in proper quantity, and of adequate quality.

The categories of information required by managers at the various levels of an organization have been described by Daniel (1961) in the following terms: (1) environmental information -- relating to the social, political, technical and economic aspects of the environment in which the organization is operating; (2) competitive information -- describing the performance, plans, and activities of other competing organizations; and (3) internal information -- concerning the operations of one's organization.



The role of information system is to acquire information in each of these categories, process the information according to requirements laid down by those who use it, transmit and report the information to those who need it, and store the information, when necessary, in a form appropriate for later reference.

Individuals and organizations are constantly bombarded by information. Some of this information arises within the organization and relates to familiar aspects of its operation and management. Another source of the information is from the environment in which the organization is operating. Usually it has more variety and it is more extensive than the internal information. The number of sources, the volume of information, and the variety of subjects to which the information refers are therefore considerably greater in the external environment.

The task of information gathering is similar to the piecing together of a complex puzzle where the parts that are available may refer to that or to one or more other puzzles with the same general characteristics. The task involves not only looking for the pieces of the particular puzzle of immediate concern but also deciding whether a piece that is picked up actually belongs to that puzzle or to one of the others.

The required activities of the acquisition function in the strategic information system are, therefore: (1) to gather as much information as possible (within the time and resource constraints) concerning subjects that are relevant to organizational decision making; (2) to recognize subjects that are relevant, so that the chances of neglecting information that is important to the organization are minimized; and (3) to gather the information required with a minimum of distortion.

On the other hand, the purpose of the interpretation function in a strategic information system is to assess the significance of the information that has been acquired. This significance may be apparent from the magnitude of the changes and trends that are detected by comparison of the new information with that which has been gathered previously. One of the most important characteristics of the successful interpreter of strategic information is the ability to integrate a vital piece of new information into a background that has been previously available.

The best results and economy of effort are usually obtained by close and continuing cooperation among managers and subordinates. The interpretation placed on strategic information is subject to the personal characteristics of the individual making the evaluation. Each interpreter brings to the task his own subjective value systems, perceptions, experiences, and judgement. Individual interpretations can, therefore, be expected to be biased to some degree.

Interpretation of information depends to some extent on the situation in which the evaluation was made. For example, in some circumstances it has been observed that less information was needed to conclude that a favorable event would occur than to conclude that an unfavorable event would occur. The desirability of the event to the individual concerned may, therefore, affect the amount of information that is required to reach a conclusion. In the same way, it has been suggested that more information is required to change previous conclusions than to establish that conclusion in the first place. Information that is inconsistent with that collected earlier is likewise often rejected.

The interpretation function is one in which intuition and creativity play an important part. Interpretation requires an emphasis on communication, the interpreter must avoid behavior that may distort the message. For

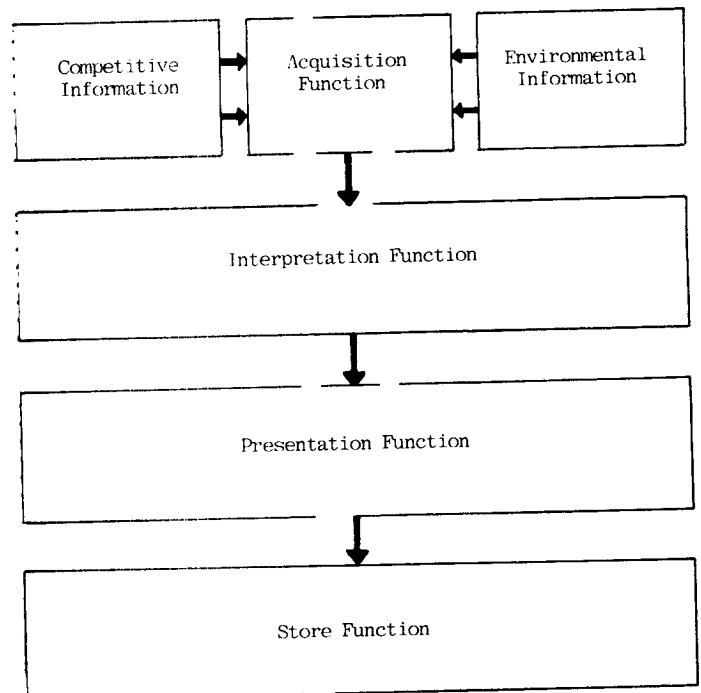
example, it has been shown that persons passing information often exhibit a desire to simplify the message. The interpreter must be aware that simple words often have a number of different meanings.

Unless these guidelines are scrupulously observed, the result may be that the most vital and significant strategic information will go undetected.

The presentation function in the strategic information system is concerned with making the information available to managers and others responsible for decisions. A way must be found of bringing the details of the information available to the attention of the users in a manner designed to attract their interest. The chosen method must also provide users with the least possible diversion of effort from their other tasks and responsibilities.

The most important general aids to presentation of the content of a store of strategic information are a series of comprehensive directories and catalogues and competently compiled thesaurus. The aim in this most general method of presentation of the data is to display the contents in a form that is of immediate assistance to users in their search for information.

Establishment and maintenance of easy access to information is a first and continuing objective of the presentation function of a strategic information system. This objective can be achieved by designing the necessary search and retrieval procedures so that they serve the purposes of the users rather than those of the system itself.



The strategic component of the overall information system has often been neglected in the face of the pressing need to meet the apparently most urgent requirements of the internal component. Future development however, can hopefully lead to a more balanced approach to information system design once the benefits of the strategic component are understood by top managers.

SELECTED ANNOTATED BIBLIOGRAPHY

- Ackoff, Russell L., "Management Misinformation Systems," *Management Science*, Vol. 14, 1967, pp. 147-156.
- Burch, John G. and Felix R. Strater, "Tailoring The Information System," *Journal of Systems Management*, February 1973, pp. 34-38.
- Daniel, Ronald D., "Management Information Crisis," *Harvard Business Review*, September-October 1961.
- Gorry, Anthony G. and Michael Scott Morten, "A Framework for Management Information Systems," *Sloan Management Review*, Vol. 13, Fall 1971, pp. 55-70.
- Katz, Daniel and R. L. Kahn, *The Social Psychology of Organizations*. New York: John Wiley and Sons, 1966.
- Keen, Peter G., "DSS: An Executive Mind Support System," *Datamation*, November 1979, pp. 117-122.
- Keen, Peter G. and Michael S. Morten, *Decision Support Systems: An Organizational Perspective*. Reading, Mass.: Addison-Wesley Publishing Co., Inc. 1978.
- King, William R. and David I. Cleland, "Decision and Information Systems for Strategic Planning," *Business Horizons*, April 1973, pp. 29-36.
- Litecky, Charles R., "Corporate Strategy and MIS Planning," *Journal of Systems Management*, January 1981, pp. 36-39.
- Mc Lean, E. R. and J. M. Soden, *Strategic Planning for MIS*, New York: John Wiley and Sons, 1977.
- Naylor, Thomas H., *Corporate Planning Models*. Menlo Park, Calif.: Addison-Wesley Publishing Co., Inc., 1979.
- Radford, K. J., *Information Systems for Strategic Decisions*, Englewood Cliffs, N. J.: Prentice-Hall Co., 1978.
- Richard A. Johnson, Fremont E. Kast, and James E. Rosenzweig, *The Theory and Management of Systems*. New York: Mc Graw-Hill, 1978.
- Schendel, Dan E. and Charles W. Hofer, ed. *Strategic Management*, Boston: Little, Brown, 1979.
- Stiles, Curt, "The Advent of Strategic Information Systems," *Managerial Planning*, September-October, 1980.
- Thierauf, Robert J., *Decision Support Systems for Effective Planning and Control: A Case Study Approach*. Englewood Cliffs, N. J.: Prentice-Hall, 1982.
- Wagner, G. R. "Decision Support Systems: Computerized Mind Support for Executive Problems," *Managerial Planning*, September-October 1981, pp. 3-8, 16.
- Zachman, John A., "The Information Systems Management System: A Framework for Planning," *Data Base*, Winter, 1978.
- Zanvetas, Z. S., "Towards Intelligent Management Information Systems," *Industrial Management Review*, Vol. 9, no. 3, September 1968, p. 22.

INTRODUCTION OF A COMPUTERIZED MANAGEMENT INFORMATION SYSTEM IN A NON-SOPHISTICATED ENVIRONMENT

Tom Marshall, Nicholls State University
Bruce L. McManis, Nicholls State University
Connie Schulte, Nicholls State University
L. W. Shell, Nicholls State University

This paper examines one situation where an automated management information system is being introduced. The firm had undertaken an extensive program to introduce the system to the people who would come in contact with it. What is reported here are the expectations and concerns of the employees after the education program. The research shows that education programs can be effective in preparing a very unsophisticated workforce for the introduction of an automated management information system. The research suggests that employees' expectations were independent of their level of responsibility, while their concerns were not. Both expectations and concerns were independent of standard demographic variables.

INTRODUCTION

New pressures for increased productivity in manufacturing, distribution, services, and especially the office, together with renewed emphasis on improved quality for American products, imply greater reliance on computer assistance in the future. (1) Thus even organizations that have traditionally operated with employees with limited educational backgrounds and minimal use of information are attempting to install computerized management information systems.

Robey (3) and others have suggested that one common cause of system failures is the ignoring of users' psychological reactions. Even under ideal circumstances this is a critical area. The less sophisticated a potential user group is the more important, and difficult, it becomes to insure that they will react favorably. Presumably, beyond some point it is not reasonably possible to alter thinking enough to make a system acceptable. This study examines one situation where initial sophistication was very low and evaluates the effectiveness of the firm's education program.

This research also determines whether user expectations and concerns regarding computerization are dependent upon other factors, such as the individual's position within the organization, use of the system, and previous experience with computer systems. The findings of this research also allow for some projections to be made about the potential success of future system implementations in similar situations.

RESEARCH DESIGN

The subjects of this study are employees within one division of a major manufacturer for the oil exploration and production industry. Educational levels of employees tend to be relatively low and their computer literacy very limited. Prior to this study management had spent approximately one year attempting to inform the employees of the benefits and drawbacks of computerization.

The study is restricted to primary and secondary computer system users within this division of the firm. Its departments are largely administrative and clerical in nature, involving such functions as purchasing, warehousing, and other services. The division's administrative nature, its management's goal to convert to automated information systems, its many current and potential computer users, and its diverse employee base (according to level within the firm) make it an ideal candidate for this study.

The researchers constructed a questionnaire to identify employees' concerns and expectations regarding the automation/integration/computerization process. The questionnaire was distributed to all 130 relevant employees. Ninety-nine (99) useable responses (76% of the 130) were received.

DATA ANALYSIS

Section I of the questionnaire identified specific characteristics of the respondents. Respondents identified the depth of their previous computer experience, their direct involvement with the forthcoming system, and their sentiments toward company-sponsored system training programs. Information on age, and sex was also obtained to provide a demographic profile of respondents. Responses showed a broad range of previous computer

experience, overall approval of computerization and a strong willingness to participate in company sponsored training programs.

EXPECTATIONS

Analysis of the responses to the various subparts of Section II allowed the researchers to determine respondents' computerization expectations and concerns. A series of cross-tabulations of user expectations and concerns against various user characteristics were used to explore relationships among variables. Chi-square analysis was used to test hypotheses about the independence of relevant pairs of variables.

Respondents were generally favorable toward computerization. The majority of responses indicated a willingness to accept some minor drawbacks in exchange for the greater expected benefits of computerization. The respondents' expectations with respect to seven aspects of computerization are summarized in Table I.

Several other factors were tested to determine the homogeneity of these positive expectations. These included experience with automated systems, data entry activities, sex, age, and department. Counter to conventional wisdom in all cases expectations were found to be independent of the tested factor. Therefore, the researchers believe that the company's educational program was quite effective.

Table 1
User Expectations
(all respondents)

	Agree	Disagree	No Opinion
Creates new jobs	70%	12%	18%
Increases personal value to the firm and in the job market	99%	1	0
Benefits outweigh restrictions	52	11	37
Procedure changes will be more difficult	36	48	16
Needed for company to be competitive	94	5	1
Increases need for cross-training	84	16	16
Lack of an integrated system increases costs and decreases productivity	73	10	17

A further examination of the data was carried out to examine the relationship between a person's position ("management" vs. "subordinate") and his or her expectations. The results are shown in Table II. Only perceived cost/benefit appeared to be dependent upon position. However, this item had the highest rate of nonresponse (37%), and was the only question worded in the negative. The large chi-square value may derive from dependence of response rate on company position. The researchers conclude that expectations about computerized systems

are largely independent of the respondent's position within the organization.

Table II
Test of Independence
Respondent Position Versus Expectations

	N	X ²	P(X)
Creates new jobs	82	3.264	.1036
Increases personal value to the firm and in the job market	99	.341	.8431
Benefits outweigh restrictions	61	13.554	.0011*
Procedure changes will be more difficult	84	.353	.8380
Needed for company to be competitive	98	.847	.6548
Increases need for cross-training	91	2.471	.2907
Lack of an integrated system increases costs and decreases productivity	82	2.163	.3392

*Significant at the .01 level

CONCERNS

Respondents indicated their level of concern over potential problems with computerization by responding on a Likert-type scale to nine statements. The two areas that concerned users most were accuracy of the computer reports and level of control over input documents. Eighty-six percent (86%) of the respondents expressed some degree of concern about each. Eighty-three percent (83%) of the respondents expressed "some concern or much concern" to the item related to the increased skills required for job performance. The only other factor with eighty percent (80%) of the respondents expressing concern was availability of training manuals for newly initiated operating procedures.

Based on the analysis shown in Table III involving concern over aspects of computerization, only external control proved to be not significantly affected by the respondent's position. Concern for aspects of computerization is clearly dependent upon the respondent's level within the organization with upper level management having fewer concerns than lower employees. Thus the "concerns" had quite different results than "expectations".

Table III
Concerns versus Position

	N	X ²	P(x)
Increased skills required	95	11.697	.0029**
External control of local operations	82	.613	.7361
Increased workload demand	95	11.697	.0383**
Changes in departmental procedures	91	15.206	.0005**
Availability of Training Manuals	93	5.882	.0528*
Accuracy of computer reports	96	5.012	.0816*
Control of source documents	93	8.579	.0137**
Installation of proper lighting	84	4.805	.0887*
Sharing of computer equipment	88	5.456	.0653*

*Significant at .10 level

**Significant at .05 level

The significant differences between upper-management and subordinates centered around the respondent's concerns regarding computerization. In general, the lower level respondents were concerned with their ability to successfully utilize the system, and interestingly, their ability to control the quality of the reports generated by the system. These lower-level respon-

dents expressed a much stronger interest in supplemental, company sponsored training programs to improve their skills regarding system operations. Neither lower or upper management were particularly concerned about the ergonomics of the work stations or the required changes in operating procedures associated with departmental computerization.

Upper level management did place emphasis on the external control of local operations and the need for an overall integrated system within the company, however, these factors were also perceived as important by lower levels within the division.

The most noticeable differences in concern over computerization were those factors which were closely related to the actual use of the system. Upper management expressed significantly lower levels of concern on factors regarding the necessity of increasing job skills, the potential for increased work load, the means available for a user to increase their skills, the quality of the reports generated by the system, and the general work environment of the user.

Age, sex, and amount of experience did not seem to affect levels of concern. This parallels the results found when expectations were examined.

CONCLUSIONS

Overall response was favorable toward computerization. The users' expectations were very homogeneous and were very close to the expectations of upper management. Respondents indicated a willingness to make changes in procedures to accommodate the system and, very importantly, the respondents expressed an interest in additional training for proper system utilization.

First, persons in upper level positions exhibited less concern than persons in lower positions, while expectations about the effects of computerization do not seem to be affected by position.

Employees are very company oriented in that they showed more concern for the impacts on the organization than on themselves. They also showed a strong degree of motivation and a willingness to make the change successful.

Factors such as age, sex, and experience with automated systems proved to be independent of both expectations and concerns. This is counter to common thinking.

The results of this study are for a specific case and as such cannot automatically be generalized. Additional work in this area will be required before any general conclusions can be reached, but it would appear that when approached properly an automated system can be successfully implemented in a non-traditional environment.

REFERENCES

1. Allen, Brandt, "An Unmanaged Computer System Can Stop You Dead", *Harvard Business Review*, Nov.-Dec., 1982, pp. 77-87.
2. Alter, Steven L., "How Effective Managers Use Information Systems", *Harvard Business Review*, Nov.-Dec. 1976, pp. 97-104.
3. Robey, Daniel, "User Attitudes and Management Information System Use", *Academy of Management Journal*, Vol. 22, No. 3, 1979, 527-538.
4. Swanson, E. Burton, "Management Information Systems: Appreciation and Involvement", *Management Science*, Vol. 21, No. 2, Oct., 1974, pp. 178-188.
5. Taggart, William M., Jr., *Information Systems: An Introduction to Computers In Organizations*, Allyn and Bacon: Boston, 1980.

AN INDUSTRIAL COMPUTER LITERACY PROGRAM
Connie D. Lightfoot
Assistant Professor of Information Science
Taylor University

Abstract

A series of three courses aimed at both management and labor has been taught at Fisher-Body Corporation (a GM division) by Taylor University. The courses have incorporated hands-on time on a VAX 11-750, a computer literacy telecourse, a supplemental lecture by the university professor and extensive time with a micro. The blend of techniques has yielded popular courses.

Computer literacy has emerged very rapidly as an essential component of widely diverse fields. At Taylor University there are currently two computer literacy courses available as well as entry-level computer science courses. A community computer literacy course has exploded in enrollment over the last year. In addition to these classes, a need for industrial computer literacy became evident.

During the fall of 1982, Fisher-Body Corporation (a division of General Motors) was proposing several major changes for the plant. They were hoping to secure a new tooling center which would require extensive computerization. An office automation system was coming which would require that office personnel also take steps toward computer literacy. A need for stand-alone workstations for departments within Fisher-Body like accounting and metal assemblies was emerging. Showing foresight, the plant manager and his associates approached Taylor University for a computer literacy course.

Structuring the course to meet the goals of the Fisher-Body executives and staying within Taylor University Information Sciences (INS) Department time constraints required the use of several components. A telecourse was used to supply the major lecture material of the course. Quizzes were taken home after each tape to measure comprehension and to encourage additional study using the textbook. Midway through the term, five laboratory sessions were conducted to give hands-on experience. The course ended with a final exam and an opportunity to critique the course. Two hours of academic credit were given for completing the sequence. At the request of Fisher-Body, a festive "graduation banquet" was held. Now, each of those components will be explored in more detail.

The telecourse component was "The Computer Programme" based on a television series produced by the British Broadcasting Corporation. There are 10 twenty-five minute tapes dealing with many introductory computer concepts:

Tape 1 - "It's Happening Now"

This tape provides a history of computers and shows the tremendous strides made in this field during the past three decades. Some application programs are described and micro-computer operation is introduced.

Tape 2 - "One Thing After Another"

This tape illustrates that a computer program is simply a series of instructions.

Tape 3 - "Talking To A Machine"

This tape discusses computer languages historically and currently and introduces the concept of subroutines.

Tape 4 - "It's On the Computer"

This tape explains computer storage, ROM, RAM, cassette tapes and floppy disks. It also introduces variables.

Tape 5 - "The New Media"

This tape discusses the capabilities of computers for transmitting, storing, and manipulating information.

Tape 6 - "Sounds and Moving Pictures"

This tape explores the world of computer-generated sounds and visuals.

Tape 7 - "Let's Pretend"

This tape discusses the simulation aspects of games, accident investigators, weather forecasters and other applications.

Tape 8 - "The Thinking Machine"

This tape explores the ability of the computer to collect information, analyze it, and use the information to improve decision making.

Tape 9 - "In Control"

This tape discusses the use of microprocessors to control or run various machines, from the washing machine to the family car.

Tape 10 - "Things to Come"

This tape explores the future as we advance into the computer age.¹

The tapes were used at the Fisher-Body plant to introduce a breadth of topics difficult to portray by traditional lecture methods. This method also made it possible for students to make up any missed class time by merely viewing the tapes. Because of the industrial setting, students were frequently either traveling or vacationing and they used the tapes often. A professor and a teaching assistant brought the tapes to the plant on a rotating basis so that the students saw the class professor at least every other class period.

The five laboratory sessions were the most popular part of the course. The two-hour session at the computer together promoted a great camaraderie among the class members. The mix between management and labor was viewed with great enthusiasm by the executives at Fisher Body. The lab sessions accommodated 20 people at a time using the VAX 11/750. These sessions were under the close supervision of an experienced teaching assistant who was assisted by five undergraduate computer science majors. The curriculum of the lab sessions consisted of exploring the use of software and the beginning stages of BASIC programming. The students learned to use a text editor, enter prewritten programs and, finally, create and enter a program. The natural errors and confusion were offset by the "good time" experience at the terminals. The basic purpose of the laboratory session was to overcome any lingering fears of a computer and was clearly accomplished. After each of the five laboratory sessions, the company paid for all class members to have dinner at the Taylor Dining Commons. This, too, heightened the group bonding. The after-dinner entertainment was a guest lecturer from the faculty. These optional lectures were well-attended and addressed such topics as: choosing a home computer, an introduction to artificial intelligence, and telecommunications.

The course was graded mainly by the quizzes over the videotapes and the required lab assignments. The final exam

¹Alton D. Roberts, et. al., *The Computer Programme*, (PMI Publications, Illinois, 1983), p. 17.

was approximately 20% of the grade. The final exam was comprehensive in nature and produced a great deal of stress in these non-traditional students. Many of them had not been in school for 30 or 40 years! With a great deal of encouragement and a relaxed atmosphere, the exam was a success.

After the final exam, the students were asked to fill out a critique of the class. The ratings were extremely high, yet yielded several valuable suggestions which were implemented the following year.

The "graduation banquet" was an exciting capstone. Several of the students had never graduated before (from anything!) and even brought families and friends to see the historic event! Supervisors and their underlings had great fun speculating on who had the higher grade. The atmosphere was one of success. The professor and teaching assistants had prepared computer-generated diplomas to award after the dinner. The plant manager spoke to the students concerning the use of their newly acquired skills. It was a fine ending to the first computer literacy course for Fisher-Body.

After receiving many requests from literacy class members, Fisher-Body decided to ask Taylor University to offer a second class in BASIC programming. It was intended to be an intensive three-week seminar (4 hours/week) which had the computer literacy class as a prerequisite. The Marion plant had seven Apple 2e computers in a training area. The one hour of lecture was sandwiched between two one-hour lab sessions allowing fourteen people to take the class at once. (Half of the class had lab before lecture--half after the lecture.)

The content of this course was additional concepts, computer terminology and BASIC programming. Because of the nature of programming, introductory programming concepts are very difficult to internalize for some students. The surface concept of entering a program, compiling it and executing it are easily accomplished, but the understanding of HOW to write the actual BASIC code seems to unfold on a very individual basis. The small class made it possible for the professor to deal individually with the students. Some of the students were using files, arrays and character string manipulations readily at the end of the seminar. Others were still struggling with loops, IF statements and I/O. The class attitude was a little less enthusiastic because the burden of learning was now on THEIR shoulders. The class professor did a great deal of encouraging and reiterating the progress made by individuals.

No grade or credit was given for this course. It was treated as a seminar--learn all you can. Again, the company paid for the course and book for each class member.

The benefits of these courses to Taylor University have been many. During each of the five lab sessions for the literacy class, at least five students were hired to assist. The students enjoyed the contact with adult learners and relished playing the role of teacher for them. The class students responded very well to the university students and, in fact, formed personal relationships with some of them. The university students joined the literacy class students for dinner to build cohesion within the group. Several of the students (15 to date) were later offered summer work opportunities at the Fisher-Body plant utilizing their computer skills.

The INS Department's Modeling and Simulation class has benefitted as well. Through the computer literacy class members (including most of the management team), an invitation was extended to do some simulation work for the plant. Three modeling and simulation group projects have saved Fisher-Body valuable time and yielded needed results. The students who participated were fortunate to have a REAL industrial problem to solve. They also gained experience from the final presentations given to Fisher-Body managers.

The obvious benefits to Taylor of high public visibility and financial help also were realized. Many of the class members had only a vague idea that Taylor University was "somewhere" in the area. We received several potential students from parental participation in the class. Since the classes were treated as continuing education classes, the supervising department received a percentage of the income

from each class, which helped meet current hardware demands.

The teaching benefits were a pleasant surprise for the professor. These non-traditional students were very eager learners. They were highly motivated because of future job potential. It was a joy to work with people who really wanted to learn about computers. The relaxed atmosphere was also a pleasure. Fisher-Body was emphatic in stressing the main goal of relieving people of their fear about computers. The course moved leisurely through the material with lots of opportunities for questions in order to meet this goal.

Our experiment in industrial computer education is continuing. We are refining the two classes and adding a third. We look forward to approaching other corporations with a similar plan. The need for computer literacy will continue for at least a few years, and Taylor University is working to meet that need.

HIGH TECHNOLOGY, EMPLOYEE DISPLACEMENT,
AND SOCIAL RESPONSIBILITY

Michael Bisesi
Assistant Dean
College of Business Administration
University of Houston

ABSTRACT

The latest advances in information systems and other areas of "high technology" are promoting greater levels of economic productivity. But these same technologies are also contributing to the displacement of many workers, and to the "de-skilling" of many of those who remain. This paper considers the relationship between technological change and employee displacement, and presents some possible remedies for addressing this problem in the context of social responsibility.

To what degree should business or government (at any level) be responsible for the retraining of technologically displaced workers and for placing them in jobs suitable for their new talents?
(Steiner and Steiner, 1980, p. 163)

We are often guilty of "frozen evaluation." Once we have found cozy niches for people, organizations, or societies, we fix those descriptions in our minds as immutable reference points. But we are not permitted that satisfaction for very long. As educators, we have a unique opportunity to address a variety of changing conditions, but we also have a unique problem. The latest advances in information systems and other areas of "high technology" (including scientific instrumentation, fiber optics, chemicals, and related fields) promote greater levels of economic productivity (not to mention jobs for us and for our students). But these very same technologies are also contributing to the displacement of many workers, and the "de-skilling" of many of those who remain. This paper considers the relationship between technological change and employee displacement, and presents some possible remedies for addressing this problem in the context of social responsibility.

HIGH TECHNOLOGY AND EMPLOYEE DISPLACEMENT

While "high tech" has been trumpeted by many as an employment savior of the future, the typical job change in this decade appears to involve losing a job at a steel mill and finding work in a bar or restaurant (Mollison, 1985). The Commerce Department's 1985 U.S. Industrial Outlook reports that even with the growth of the service sector and "knowledge" industries, employment has remained basically unchanged.

Are we destined to be a nation of McDonald's workers and insurance salespersons, is "high tech" going to transform our society, or does the truth lie somewhere in between? It is quite clear that our traditional manufacturing industries can remain competitive, and thus maintain jobs, by supplanting automated processes for people. This paradox is compounded by the technologically-induced shift in employment requirements from physical strength to cognitive skills (Galambos, 1984).

One source of the paradox lies in the nature of the work force--our "dead end labor" problem (Reich, 1983). While workers have been cast as an abstract labor "pool" by our macroeconomic policies, and only the "truly needy" (rather than the entire labor force) have access to social welfare benefits, it is little wonder that more Americans are unemployed, underemployed, or trapped in "dead end" jobs instead of finding new opportunities in "high tech" industries (Reich, p. 201). Because social programs such as job training, health, housing, and nutrition have often been viewed as charitable programs of last resort, we have been unable to prepare the work force to meet the demand that "high technology" has placed on the economy (Reich, p. 206).

And the problem is likely to worsen. Even skilled "brain workers" stand to lose out in a setting based on microelectronics. Productivity increases resulting from technology probably will not produce an adequate number of jobs to replace the technologically-displaced positions. Ironically, more growth is forecast in service and clerical jobs than high technology occupations. Thus, jobs at the lower end of the skill continuum will provide many more opportunities than those at the higher

end, and the continued growth of high technology is likely to compress rather than expand the range of skills needed for employment (Rumberger, 1984; Rumberger and Levin, 1984; Levin and Rumberger, 1983).

Why should we be concerned about this technologically-induced shift in employment? There are those who recall with great optimism how the workforce has always adjusted to changes, whether because of new agricultural implements or new methods of manufacturing textiles or improvements in printing, and "high technology" is just (they contend) another chapter in that epic story of rugged individualism and success.

But this is not just another chapter. The ubiquity of technology, the quickened pace of change, worker "de-skilling," and the continued absence of a national education and training strategy make this episode a very different one (Choate, 1983). An increase in international competition, greater specialization, and a pronounced drop in the quality of our educational system have all contributed to structural unemployment (Samuelson, 1983). Compared to the past, structural unemployment is more pervasive, there is a greater disparity in the skills required as a result of technological advancement, and neither government nor industry seems to know how to prepare workers for these (and future) jobs (Samuelson, p. 22).

Depending on whose figures one uses, structural unemployment ranges from 100,000 to 3 million dislocated workers. This figure will probably increase, if a study published by the National Science Foundation is at all correct. Computers, robots, and other advanced technologies may mean that "11 million fewer workers would be required by 1990 and 20 million fewer by 2000 to produce the same goods and services that would be delivered under the no-changes scenario." The year 1990 may also see 900,000 fewer sales workers, 2 million fewer managers, and 8 million fewer clerical workers, all largely attributable to technological advancements (Corrigan and Stanfield, 1984).

The cost of "high technology" to people is substantial. "You can't," said an unemployed steelworker, "train someone 50 years old for computers (Samuelson, p. 22). "Nobody," remarked an ex-assembly line worker, "is beating down your door to give you a job no matter how much training you get" (Corrigan and Stanfield, p. 252). People are losing good jobs, and permanently. And the ripple effect, from schools to supermarkets to suppliers in allied businesses, is substantial.

The social consequences of this sort of unemployment were calculated by a Johns Hopkins professor of public health, who estimated "each percentage point of national total unemployment sustained over six years accounts for 37,000 deaths--including 920 suicides, 650 homicides, and 500 total cases of cirrhosis of the liver," as well as "4000 admissions to state mental hospitals and 3,300 incarcerations in state prisons" (Corrigan and Stanfield, pp. 256-257).

Obviously, there still will be non-technological jobs in the future. Transportation of raw materials, goods and products, ranging from coal to grain to steel, will not occur via satellite or ethernet, but by ships, trains, trucks and barges. The energy and defense industries, though becoming more technological, are still production-oriented, and will likely remain so--for now (Etzioni, 1984). But "high technology" will be a potent force in shaping future employment.

SOCIAL RESPONSIBILITY

The concept of "stakeholders" has an important role in determining responsibility for technologically-displaced workers (Sturdivant, 1981, p. 11). Business, in many ways, is a quasi-public entity. Business employs and does business with citizens whose interests are represented by a government. Stakeholders, those who are affected by and who feel they have a "stake" in a business, often are affected by business decisions that may not have been considered to be of public interest. Thus, both business and government share responsibility for this problem.

Business and government have tried a number of ventures to enhance opportunities for the workforce. Many states have tried on their own, ranging from the Bay State Skills Corporation in Massachusetts, to South Carolina's "Special Schools" that train employees for new companies, to similar programs in Michigan, Minnesota, Pennsylvania and Texas (Pilcher, 1983). California's Family Economic and Security Act merges welfare and training programs so that people receive a training stipend instead of a welfare check (Riffel, 1983). Under the Federal Job Training Partnership Act, "laid-off workers will be shown how to prepare resumes and sharpen their job-seeking skills and also will be provided on-the-job or classroom training for new jobs." Moreover, businesses "will be given information on how to apply for government contracts," thus helping workers in a direct way but also working at "long-range economic development needs" ("Retraining Grant," 1984).

The Job Training Partnership Act has been heralded by the Administration as a way for business and government to join together in the cause of economic opportunity. Unfortunately, "there is little evidence available as to the employment impacts on training participants." As a worker ages, there is a "deterioration in the unemployment advantage attributable to the extra years of education" (Blakemore, 1984).

And confusion seems to be the order of the day when it comes to public policies related to education, training, and employment. College students must now rely more on loans instead of financial aid grants (Millard, 1984). The Administration's proposed budget for Fiscal Year 1985 not only suggested cuts of \$330 million in what remained of student financial aid, but also proposed level-funding the Job Training Partnership Act (Saunders, 1984). Congress only reluctantly and belatedly restored section 127 of the Internal Revenue Code, the section that exempted employee educational reimbursements from taxable income.

THE NEED FOR A NATIONAL POLICY

A number of proposals have been advanced to address these issues in a more comprehensive way. A study prepared for the Joint Economic Committee of Congress recommends mandatory listing of job vacancies with the U.S. Employment Service, job search and relocation benefits, advance notification of layoffs and plant shutdowns, and a redirection of the Job Training Partnership Act to deal with long-term displaced adult workers---the very people most susceptible to high technology changes (Podgursky, 1984). Income support programs limited job search assistance, basic education for under-educated and unemployed adults and tuition assistance combined with unemployment benefits are other possible remedies (Pilcher, p. 12).

There have been specific proposals to provide on-the-job training vouchers (funded by government and business) and retraining vouchers. Changes in the tax code could allow a company to reduce its tax burden by retraining older workers for new jobs and by keeping businesses in their current locations. Unemployment insurance rates for business could reflect how often companies foist their people on the public dole. Companies could administer government-funded social services, thus enabling a greater investment in human capital (Reich, pp. 240-248).

Some of these ideas found a legislative home in the proposal for a federal Council on Economic Competitiveness and Cooperation, which was introduced in the 98th Congress (S. 2795, "Economic Competitiveness and Cooperation Act"). The purpose of the bill was "to improve long-term employment opportunities in the United States" by, among other things, reforming unemployment insurance systems in favor of training, providing loans for adult job training, providing community service employment in areas with declining industries, and providing

counseling and placement service to displaced workers (pp. 7-8).

OUR ROLE AS EDUCATORS

Some of these proposals may be too late for some workers, but nothing will change unless educators and the educational system change. Adaptability, switching to new career paths, and a strong general (rather than vocational) education "is the best insurance for survival in a time of rapid change" (Galambos, p. 39).

Some editorial writers in Britain have noted that the "British workforce is badly trained because it is by then too old to educate." Instead of technical training, young people "should be made to think, and be told how to demand as many skills as they can get," which suggests "more general education earlier, not more technical training which they have not been educated to want or absorb" ("Too Young To Train," 1984). Other writers, closer to home, support this view:

People who are vocationally trained to unquestioningly perform a single task are manifestly unprepared to design their own work, participate in decision making, assume control over their own working condition, work as members of a community of equals, or take responsibility for the quality of their own work when a boss is not looking over their shoulders. Like narrowly trained managers, these workers feel easily threatened by change and act defensively, inflexibly, and in ways society deems irresponsible when circumstances require them to adapt" (O'Toole, 1981, p. 182).

The responsibility, then, for addressing the needs of technologically displaced workers belongs to businesses, government, and education. There are those who argue that the "invisible hand" of the "free market" should remain unfettered. Quite frankly, though, we really have a choice "between a covert politics that stymies economic and social progress and an open politics that promotes it" (Reich, pp. 273-274).

REFERENCES

- Blakemore, A.E. "Human Capital Investment and the Reduction in the Unemployment Rate Consistent with Nonaccelerating Inflation." Quarterly Journal of Business and Economics, Vol. 23, No. 2 (Spring 1984) pp. 3-14.
- Choate, P. "Technology and the Workers." In High Technology: Public Policies for the 1980s, edited by Richard S. Frank. Washington, D.C.: Government Research Corporation (1983) pp. 89-91.
- Corrigan, R. "Choosing Winners and Losers." In High Technology: Public Policies for the 1980s, edited by Richard S. Frank. Washington, D.C.: Government Research Corporation (1983) pp. 6-42.
- Etzioni, A. "The Two-Track Society." National Forum: The Phi Kappa Phi Journal, (Summer 1984) pp. 3-5.
- Galambos, E.C. "A 'High Tech' or a Service Economy Future?" National Forum: The Phi Kappa Phi Journal (Summer 1984) pp. 38-39.
- Levin, H.M., and Rumberger, R.W. "The Educational Implications of High Technology." Institute for Research on Educational Finance and Governance, Stanford University (1983).
- Millard, S. "Student Financial Aid: From Grants to Loans." State Legislatures (October 1984) pp. 11-17.
- Mollison, A. "Job Changes in the 80's Lead Workers from Steel Mills to Restaurants." Houston Chronicle (January 6, 1985) Sec. 1, p. 6.
- O'Toole, J. Making America Work: Productivity and Responsibility. New York: Continuum (1981).
- Pilcher, D. "Job Training in the States." State Legislatures (June 1983) pp. 9-14.

- Podgursky, M. "Labor Market Policy and Structural Adjustment." In Policies for Industrial Growth in a Competitive World. Washington D.C.: Joint Economic Committee, Congress of the United States (1984) pp. 71-96.
- "Retraining Grant to Aid Ex-Texaco Employees." Houston Chronicle, (November 5, 1984) Sec. 1, p. 12.
- Riffel, R. "Job Training: A New Opportunity." State Legislatures (June 1983) pp. 6-8.
- Rumberger, R.W. "High Technology and Job Loss." Institute for Research on Educational Finance and Governance, Stanford University (1984).
- Rumberger, R.W., and Levin, H.M. "Forecasting the Impact of New Technologies on the Future Job Market." Institute for Research on Educational Finance and Governance, Stanford University (1984).
- Reich, R.B. The Next American Frontier, New York: Times Books, 1983.
- S. 2795, "Economic Competitiveness and Cooperation Act." Senate bill introduced in 98th Congress, 2nd Session (1984).
- Samuelson, R. J. "The Old Labor Force and the New Labor Market." In High Technology: Public Policy for the 1980s, edited by Richard S. Frank. Washington, D.C.: Government Research Corporation (1983) pp. 22-29.
- Saunders, C.B. "Many Cuts Proposed in Student Aid." Higher Education and National Affairs (February 10, 1984), pp. 1, 5.
- Steiner, G.A., and Steiner, J. F. Casebook for Business, Government, and Society. 2nd Ed. New York: Random House (1980).
- Sturdivant, F. Business and Society, Revised Edition. Homewood, Illinois (1981).
- "Too Young To Train." The Economist (September 8, 1984) p. 18.

THE DESIGN AND ANALYSIS OF THE INFORMATION SYSTEMS CURRICULUM

Dr. Karen A. Forcht, Assistant Professor
Information and Decision Sciences Department
School of Business
James Madison University

Abstract

This presentation includes elements of design of the Information Systems Curriculum, as well as evaluation and analysis of an on-going curriculum. Programs must be designed to give the student a broad-based background beyond manual skills with a high emphasis on personnel administration, labor relations, and new technological equipment found in the business office. If our current curricula is to reflect the business environment more directly and prepare students for future employment in an office environment, the Information Systems Program needs to be revised to adequately incorporate what is currently happening and what is anticipated for the future in terms of information needs.

"Business organizations are now beginning to recognize the importance of viewing records management, data processing, word processing, and administrative services as integral parts of information processing or information management." (3) In many schools, business administration curricula as yet do not reflect any integration of these activities, but are treated as completely separate curricula. (3)

A consequence of this separation is that data processing is considered a business administration subject and is usually found in the business core requirement, while the other three subject areas are studied only by secretarial or office administration students.

In order to evaluate the information systems curriculum, we must first define the terminology presently being used in today's information environment.

"Information management is a concept that encompasses the total business organization and the Business Education/Administration curriculum. It must reflect the importance of managing information--as it already recognizes the importance of information itself." (3)

A second author gives several possible explanations for office automation (5):

1. "Office automation is a merging of technologies and an integration of functions providing quality information and communication facilities designed to improve professional and managerial productivity."
2. "Office automation is machine-aided creation, communication, storage, retrieval, control, and disposition of information handled by professionals and work staffs in the office environment."
3. "Office automation improves both the efficiency and the effectiveness of white-collar workers. While word processing typically promotes efficiency--more typewritten pages per dollar--office automation addresses the needs of managers and professionals. It is primarily concerned with improving their effectiveness in conducting business."

It would appear that information systems is a concept that encompasses the total business organization in terms of "information flow". It is doubtful that anyone in the business environment would be totally immune from the "tentacles of information" reaching

them. The curricula must reflect the importance of managing information--as it already recognizes the importance of information itself.

"Data processing and word processing are simply two logical subsets of information processing--not totally diverse areas." (3) Separation of these two areas is no longer a feasible approach from either the results or the economics point of view. In the traditional college or high school classroom, unfortunately, courses dealing remotely with information systems are so structured that they emphasize the machinery used rather than the procedures required (a "how to", rather than a "why" approach). (3) To be truly relevant, the concept of information management must recognize the interaction of records management, data processing, and word processing in a logical way to achieve a complete picture.

ESTABLISHING AN INFORMATION SYSTEMS CURRICULUM

Jeffrey Prince in his article entitled, "What It Will Take to Manage in the '80's," says, "The office environment will consist of such things as advanced word processing systems, minicomputers, reprographics, micrographics, teleconferencing, video conferencing, and telecommunications--all interrelated through integrated networks. (6) The employees in today's office will have to be versatile in juggling the demands of technology, information, and people.

In order for office personnel to be prepared for such roles, educational programs must begin to recognize the urgent need to enhance their curriculums.

Suggestions for Curricular Changes

Programs must be designed to give the student a broad-based background beyond manual skills with a high emphasis on personnel administration, labor relations, and new technological equipment found in the business office.

"Instructors must incorporate more decision making and case study problems into our classes which would necessitate the use of our students' judgmental skills and imagination in seeking a solution. Assignments given with fewer instructions for implementation encourage our students to rely upon their own mental capabilities." (4)

A few suggested methods/vehicles of curricular improvement might be:

1. Design upper-level courses in which decision-making and organizational philosophy where students will encounter real-world business problems or situations in which they must seek solutions. By problem solving in the classroom, the student will have an opportunity to critically examine the consequences.
2. Purchase more advanced technological equipment for classrooms in order that learning becomes a close semblance to what is actually taking place in the business office.
3. Vary delivery systems to students. Because students are exposed to different managerial styles in the business office, our programs should encompass more than one or two teaching learning styles. Students, when they are later employed, will have to be willing to adapt to the style used in their place of employment.
4. Expose students to a wide variety of equipment. The most advanced offices possess a combination of different technological equipment on the market. A classroom filled with one name brand of equipment is obsolete.
5. Design courses in professional development. Well-established personnel firms offer seminars in job-seeking skills, job maintenance, methods of seeking a promotion, the art of seeking and achieving a desired raise, and office politics. We must extend our curricula to encompass such features. (4)

"It is up to us as educators to change the thinking philosophy of businesses regarding the capabilities of our graduates through adopting an ongoing means of updating our curriculum." (4)

Whether initially establishing or assessing an ongoing information systems program, we must use some of the ideas on the following checklist in order to ascertain whether or not our curriculum is relevant:

Checklist for Information Systems Curriculum

1. Are we upgrading our curriculum to include more courses that will sharpen our graduates' judgmental skills?
2. Are we maintaining a high level of office operations analysis that will enable us to predict what the future holds for our graduates?
3. Are we appraising the technology being used to alter office functions and then designing programs that will meet major educational goals/needs?
4. Are we encouraging internships for longer periods of time? Are we including both the skills of judgmental capacity, as well as involvement in business technology, in our internships?
5. Are we advertising/promoting our program to the community?
6. Are we actively seeking the advice/counsel of business persons when designing or evaluating our programs?
7. Are we teaching job maintenance and projection, as well as preparatory/entry-level skills? (4)
8. Are we establishing/maintaining a program that is appropriate for our community served, student populace served, employment statistics, etc.?
9. Are we arranging for support persons/materials in order to make the program more realistic/relevant?

10. Are we ensuring that high school guidance counselors are aware of the tenets of our program?
11. Do we follow up on graduates to assess whether the program is relevant? (8)

PROJECTIONS FOR THE FUTURE OF INFORMATION SYSTEMS

Marguerite Gounley observed in 1948 that:

"Business training should be as adjustable and flexible as business. It has to be constantly evaluated, improved, and revised. It is not static; it is dynamic. Because this is true, the business teacher has need for consulting with businessmen and employees in office occupations to keep the training program of the school in line with modern business methods and practice." (1)

Lois Hagan, stating her view in 1981, commented that:

"If . . . programs are to be beneficial, they must be an integral part of the community in which they exist and must reflect the daily occupational life of that community. To accomplish this goal, close cooperation between the school and business, labor, and the industrial community is essential." (2)

Amy Wohl, when speaking as the Honors Lecturer at the College of Business Administration, Oklahoma State University, in October, 1982, further extends the needs for future considerations by stating that:

"The office of the future is taking shape, here and now, all around us. When a workman was given an electric drill to drill a hole in a car body, that was an improvement over a hand-powered drill. An office parallel would have been the use of an electric typewriter to replace handwritten or manually-typed letters. Within five years, it will be close to impossible for a person to work in an office who cannot use the keyboard of word processing or data processing equipment. In 20 years, it will be unheard of." (7)

The exciting challenge Ms. Wohl sees in the current office revolution is less in the tremendous expansion of information available to individuals, than it is in bringing all the isolated units of a company together into one system. Her key phrase was "strategic office planning." (7)

Instructors must perceive the role of teacher as an active, not passive one. They need to take control of the business program and to keep the curriculum relevant.

In addition to essential teaching method competencies, instructors need to have an understanding of what is happening in the office environment.

Educators are on the threshold of an exciting educational area. As was emphasized before, information systems means the use of the computer and other sophisticated equipment to aid in the instructional process. Automation is truly changing the method all teachers will use in instructing various subject areas. Basic skills take on an exciting air of the future.

CONCLUSION

If our current curricula is to reflect the business environment more directly and prepare students for future employment in an office environment, the Information Systems Program needs to be revised to adequately incorporate what is currently happening and what is anticipated for the future in terms of information needs. Educators must be prepared to understand and demonstrate teaching methods and business knowledge that reflects these new trends.

Teaching information systems is one solution to

bridging the gap between the business classroom and the office environment. The way in which the office is operating dictates how the classroom needs to be conducting education, rather than vice versa. This probably brings up the argument of the "tail wagging the dog," but education must follow the dictates of industry as they are setting the standards that we must follow. The areas discussed today are not separate entities, but interrelated parts of one whole. The student today needs to be trained in the business classroom to go out into the office via information systems competencies.

REFERENCES CITED

- 1) Crumley, Marguerite, "A Guide for Advisory Committees in Business Education," Business Education--A Retrospection, Monograph 133, Southwestern Publishing Company, Cincinnati, Ohio, 1978. (Reprinted from The Balance Sheet, April 1948), p. 2.
- 2) Hlavac, Lois, "Involving Advisory Committees to Update Subject Matter Content," Updating Content in Secondary Business Education, National Business Education Yearbook, No. 19, Reston, Virginia, 1981, p. 36.
- 3) Johnson, Margaret, Editor, "The Changing Office Environment," The National Business Education Yearbook, No. 18, NBEA: Reston, Virginia, 1980.
- 4) Loston, Adena Williams, "New Curriculum Needed for Prospective Office Managers," Business Education Forum, December 1981, p. 8.
- 5) Owens, Bill, "Business Education Implications of Office Automation," The Balance Sheet, September-October, 1982, p. 12.
- 6) Prince, Jeffrey S., "What It Will Take to Manage in the '80's," Administrative Management, February, 1980, p. 34-35.
- 7) Wohl, Amy, Honors Lecturer for the Oklahoma State University College of Business Administration, "Stillwater News Press," Stillwater, Oklahoma, October, 1982.
- 8) Wood, Merle W., "The Teaching of Automated Data Processing in the High School," Monograph 116, Southwestern Publishing Company, Cincinnati, Ohio, 1967, p. 52-54.

Information Systems Needs for the Business Curriculum

by

John F. Schrage
and
Robert A. Schultheis

Southern Illinois University at Edwardsville
Management Information Systems Department
Campus Box 106, Building II
Edwardsville, IL 62026-1001
(618)-692-2504

ABSTRACT

Because of the growing importance of computers, the standard introduction to data processing course is no longer sufficient for all students. There is a need for a computer literacy course to provide every subject matter area with instruction in the use of the computer as a tool. Business students further need to develop computer skills in the use of major software packages in a management context. A changing need is thus becoming evident at the college level for instruction in the use of the machine as a tool. This paper examines the content of computer courses.

THE COMPUTER LITERACY ISSUE

Universities are facing increasingly sophisticated students in terms of computer knowledge than they faced five years ago. Elementary and secondary schools have been increasing the number of courses they offer and the emphasis they place on computer skills and concepts in their programs. (Hama, p. 47) The increased emphasis on computer literacy and computer skills at the elementary and secondary levels should be producing changes in the computer courses offered by universities.

Because of lack of facilities, many computer courses in the past emphasized computer concepts rather than hands-on computer skills. As universities acquire more extensive computer facilities, especially micro-computer work stations, the nature of the computer courses is metamorphosing. Large amounts of time may be devoted to hands-on use of the computer instead of discussion of computers use. Concomitantly, hands-on computer skills have increasingly become a requirement in many general education programs. In fact, the general education skill areas needed by students are being classified into (1) written expression, (2) oral communications, (3) critical thinking, (4) statistics, (5) computer programming, and (6) foreign languages.

DATA PROCESSING VERSUS INFORMATION SYSTEMS COURSE

Schools of business face an increasingly computer literate student. For such students, the traditional introduction to data processing course is not likely to be effective. The traditional introduction to data processing course is currently under review because many students have already acquired the necessary computer concepts at the high school level. Unfortunately, the introduction to data processing course is still used by many institutions to meet the American Assembly of Collegiate Schools of Business (AACSB) core requirement for information systems. (Aulgur, p. 7) However, neither training in rudimentary computer skills at the general education level nor completion of an introduction to data processing course are likely to provide the preparation in information systems needed by the business student.

Confusion over which computer concepts should be mastered by the business student may reflect confusion between the area traditionally called "data processing" and the emerging area of "management

information systems". In other words, the meanings ascribed to the terms data processing and management information systems are often undifferentiated. This lack of differentiation is reflected currently in the title, content, placement, and goals of courses designed to provide computer instruction to collegiate students majoring in business.

To liken data processing to management information systems is to liken record keeping to accounting. The important point here is that confusion regarding the meaning of recordkeeping and accounting will lead to seriously confused curriculum content, sequencing, and course goals. Likewise, confusion regarding data processing and management information systems appears to have led to poorly constructed collegiate curricula.

AACSB, DPMA, AND ACM ON INFORMATION SYSTEMS

The AACSB recognizes the need for information systems in its document entitled, Accreditation Council Policies, Procedures, and Standards. The AACSB curriculum standards require that the business content "shall be responsive to ... technological developments and shall reflect the application of evolving knowledge in ... quantitative sciences". (AACSB, p. 26) Additionally, students must be provided with the common body of knowledge (CBK) in business administration [including] ... a basic understanding of the concepts and applications of ... management information systems including computer applications". (AACSB, p. 27) While these recommendations do not require a specific course in management information systems, the implication is that all students must at least integrate instruction in systems concepts in their curricula.

The major business computer curriculum studies conducted by the Association for Computing Machinery (ACM) and the Data Processing Management Association (DPMA) place differing emphases on the content of DP and MIS. The DPMA CIS curriculum uses the introduction course (CIS-1) as a requirement and recommends an elective course in decision support systems (CIS-10). (Adams and Athey, p. 11, 24) The ACM IS curriculum includes the introduction to programming course (P-1) to provide for the technical concepts and uses the information systems in organizations course (I-3) to provide MIS concepts. (Nunamaker, p. 793-4) Neither curriculum study seems to place importance on information systems for the business student majoring in a content area other than information systems. Each study provides a

specialized program for the business information systems specialist rather than providing a program for the business major who needs information systems concepts integrated into another business field.

THE INTRODUCTORY INFORMATION SYSTEMS COURSE

Given the importance of decision support systems to the modern manager, there is a need for the general business major to complete a course which emphasizes information systems rather than data processing. While general education computer skills courses can provide instruction in basic computer skills and concepts, the computer skills must be applied within business content to provide for transfer of education. Thus, all business students should take a course in the application of information systems tools to business.

The general education curriculum of Southern Illinois University at Edwardsville, requires that all students complete a computer skills course at the general education level of their college program. The course supplants the computer literacy course for general students and the traditional introduction to data processing course for business students. Subsequently, all business majors enroll in an introductory information systems course which addresses decision support systems within the business environment as opposed to data processing concepts.

THE COMPUTER SKILLS COURSE

A computer skills course should include instruction in programming and the use of common packaged software. Student must have sufficient hands-on experiences in programming to design, write, and debug elementary programs on the computer. The rationale for inclusion of computer programming in such a course is that students should be reasonably proficient in a language skill which controls computers. (Johnston, p. 11) However, students should also develop skill in the use of packaged software in such application areas as spreadsheets, word processing, and statistics.

Specifically, each student should be able to demonstrate the following skills and knowledge upon completion of the course:

- (1) Familiarity with computer terminology.
- (2) Knowledge of the use of computers in the field of study which the student has elected.
- (3) The ability to read a simple computer program in at least one high level standard language.
- (4) The ability, given a problem statement, to chart the logic for the solution and write a program dealing with the problem on either a mainframe or microcomputer.
- (5) The ability to use a statistical package for descriptive statistics.
- (6) The ability to use spreadsheet and word processing application software on a microcomputer.
- (7) The ability to use the computer facilities available on campus.

Recommended topics, with the amount of time devoted to these topics, for the computer skills course are:

- (1) The Computer and Its Capabilities. (03%)
- (2) Hardware and Software Concepts. (05%)
- (3) BASIC Programming Concepts. (35%)
To include concepts, input/output, calculation, looping, and reports.
- (4) Overview of High Level Programming Languages. (05%)
- (5) Software Packages on the Microcomputer. (35%)
To include the use of spreadsheets and word processing software and a discussion of other major types of application packages.
- (6) Statistical Packages and their Use. (05%)
- (7) Use of the Computer in Life. (10%)
To include the use of the computer in the student's personal life, classroom, workplace,

- business and industry, science, medicine, research and design, art, entertainment, government, privacy, and crime prevention.
- (8) Future of the Computer. (02%)

The following projects may be used to help achieve the course objectives:

- (1) The reading of five computer articles relevant to the student's field and the summarization of the material using a microcomputer word processing application package.
- (2) The use of a microcomputer spreadsheet software package to complete a budget problem.
- (3) The writing and running correctly of BASIC programs, in increasing complexity, to illustrate listing, calculations, control breaks, and file processing.
- (4) The use of a statistical package, such as SAS or SPSS, to produce descriptive statistics from a set of data.

THE INTRODUCTORY INFORMATION SYSTEMS COURSE

Common practice has been to engage the computer specialist to design decision support systems and to house the decision support systems on the mainframe. Because of the rapid acceptance of the microcomputer and personnel shortages in the computer field, decision support systems are being transferred to the microcomputer and the user has become the systems designer. Thus, knowledge of information systems and, especially decision support systems, have become essential for the business student.

Instruction in information systems may be integrated into the functional business areas. However, the integration of information systems content into the functional areas is materially enhanced if students study information systems in a systematic way prior to the integration. Thus, a course in information systems, emphasizing decision support, should be required of all business majors. (Manning, p. 42) Moreover, the key point is that such a course should occur early enough in the curriculum that students may transfer the skills and knowledge learned to the decision problems they encounter in the functional courses.

The information systems course should provide the student with the acquisition of the following abilities in information systems skills and knowledge:

- (1) Describe and use basic concepts of systems analysis.
- (2) Explain how computer system and computer personnel resources can be used by the manager.
- (3) Describe user involvement in the information system cycle.
- (4) Describe and use common tools of systems analysis.
- (5) Describe computer decision support systems useful to the manager.
- (6) Use common decision support software to solve management problems.
- (7) Describe the ethical and social considerations needed to develop and use information and decision support systems.

To achieve these objectives, the content of the information systems course should include the following major topics:

- (1) Managing information with computer technology. (05%)
To include management and systems concepts and terminology.
- (2) Information, decision support, and expert systems. (20%)
To include the study of the nature and impact of information on decisions.
- (3) Tools and techniques of systems analysis and design. (05%)

REFERENCES

- To include charting, data flow diagrams, interviews, and output design.
- (4) Project management and implementation. (15%)
To include scheduling, cost benefit analysis, and system resources.
- (5) Traditional and structured systems. (20%)
To include the life cycle approach and analysis-design techniques in both structured and nonstructured modes.
- (6) Decision making using computers. (20%)
To include use of spreadsheet, data base, and project management programs.
- (7) Management and societal implications of information systems. (10%)
To include the study of computer system controls, security, and standards and the social and ethical aspects of information systems.
- (8) You, information systems, and the future. (5%)
To include a study of the student as a user, programmer, and analyst in a rapidly changing environment.

To develop experience in the use of computer for managerial decision making, decision support projects such as the following might be used:

- (1) A project management activity using associated project management software such as Project Planner or Visischedule on a microcomputer.
- (2) A cash flow problem and a lease/purchase comparative cost problem using spreadsheet software such as Visicalc, Multiplan, or Lotus 1-2-3 on a microcomputer.
- (3) A data base project using relational data base software such as dBase II or III, on a microcomputer or IDMS/R or other relational data base on a mainframe.
- (4) An extraction or management report using fourth generation languages such as MAPPER, LINC, EXPRESS, or IFPS on a mainframe/mini computer or IFPS or FOCUS on a microcomputer.

Computer decision support projects may be completed using teams of students with different business specializations to simulate real-world project team composition and to provide some depth in the application area used for the project.

Additionally, case situations pertaining to the other topics in information systems described above should be included. (Pipkin, p.59) It is suggested that written reports of the analysis of these cases be prepared using word processing software. The written reports should be prepared in a manner consistent with the expectations of management within an organization. (Sides, p.37)

SUMMARY

The introductory computer skills course should focus on the skill of using the machine as a tool. This means that the computer literacy course should change from a topical discussion of computers in society to a hands-on course using campus computer facilities. The introduction to data processing course emphasizing the traditional topics of data processing and programming designed for data processing majors should be redesigned as a management course for all business majors emphasizing the use of software for decision support. Additionally, the integration of the computer as a decision support tool in the functional business courses and the policy course should be pursued with vigor.

Note:

The syllabus, cases, projects, and handouts used by the authors for the courses described may be obtained by requesting these materials in writing.

American Assembly of Collegiate Schools of Business. Accreditation Council Policies, Procedures, and Standards. Saint Louis, Missouri, 1984.

Achey, Thomas H. "Information Gathering Skills". Interface, 1:6-11, Winter 1982/83.

Atney, Thomas H. and David R. Adams, editors. DPMA - A Curriculum for Undergraduate Computer Information Systems Education. Park Ridge, IL: Data Processing Management Association, 1981.

Baliga, Venkata and G. Daryl Ford. "Profile of the Marketing Information Systems Course in AACSB-Accredited Institutions". Interface, 4:4-8, Fall 1984.

Dickerson, Gary W. and Ralph H. Sprague, Jr. A Short History of the Information Systems Area: An Academic Perspective. Paper presentation at the First International Conference on Information Systems, Philadelphia, PA, December 1980.

Friedman, Linda W. and Hershey H. Friedmann. "An Effective Method for Teaching Ethics in a Computer Course". Interface, 6:61-62, Spring 1984.

Gama, Paul and Roy Lee. "The Impact of Computers in Secondary Education: part 1 and 2". Interface, 4:40-51, Fall 1982 and 4:58-62, Winter 1982/83.

Johnston, Randy. "Topics and Tools for the Introductory Course". Interface, 6:10-13, Spring 1984.

Manning, William A. "Decision-Making: How a Microcomputer Aids the Process". Interface, 5:42-46, Winter 1983-84.

Nunamaker, Jay P. and others, editors. "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs -- A Report of the ACM Curriculum Committee on Information Systems". Communication of the ACM, 25: 781-805, November 1982.

Pipkin, Jar. "Cases Add New Dimension to Information Systems Instruction". Interface, 6:58-61, Spring 1984.

Ralston, Anthony and Edwin D. Reilly, Jr. Encyclopedia of Computer Science and Engineering, second edition. New York: Van Nostrand Reinhold Company, 1983.

Rushinek, Sara F. "Computer Literacy for Managers". Interface, 5:28-30, Spring 1983.

Sides, Charles H. How to Write Papers and Reports About Computer Technology. Philadelphia, PA: Institute for Scientific Information (ISI) Press, 1984.

SELECTED GUIDELINES FOR EFFECTIVE COURSEWARE AUTHORING

Mr. Rick David Stuart
Assistant to the Director,
Academic Computing
Xavier University
Cincinnati, Ohio

Abstract: The following article deals with the methodology required to design and develop effective computer courseware. Examining the development process associated with the overall construction of courseware, practical suggestions are given, aimed at reducing the amount of time needed to produce quality lessons. Directed toward the instructor considering courseware design for the first time, this article suggests a common sense approach to courseware construction.

Until recently the production of information for the educational process has depended almost entirely on the written word. With the arrival of the computer in the second half of this century, however, this common standard has been challenged. As a legitimate tool for the dispensing of quality education, the computer has fostered a wide range of tools. One particular tool, the courseware authoring system (CAS), has proved itself very effective in the presentation of a wide variety of subject materials in innovative and imaginative formats. This notwithstanding, many educational institutions currently capable of supporting a CAS oriented environment fail to do so largely due to the excessive development times required in the production of CAS lessons of professional quality. An examination of the development process, however, suggests many practical considerations can be applied to reduce the amount of time necessary to construct viable CAS lessons while enhancing the quality of the lessons as a whole. Four such guidelines: 1.) Determining Lesson Scope, 2.) Defining Authoring Responsibility 3.) Selecting A Lesson Flow and 4.) Developing Effective Graphics are discussed briefly below. At the outset the lesson designer must properly determine the scope of his CAS lesson. Many confuse this with a question of physical lesson size. While size is evidently one element in lesson structure, scope here refers to the rationale behind the lesson's introduction, the audience the material is intended for, and the final form it will take on. Understanding the importance of these concerns is necessary to avoid the frustration of extensive design revisions later on. Considerations of scope force the instructor to identify the actual target group: in addition to age factors, for example, he must decide what will be the maximum level of difficulty the users will be capable of handling and what prior knowledge or exposure to related materials will be assumed in this lesson? Likewise, will the lesson ultimately stand independent of other designs or rather as one of several joint projects. On the basis of answers given to the above questions, it must then be decided what presentation vehicle will be best suited to the lesson's lesson's defined audience, i.e. dialogue, drill & practice, tutorial, or simulation? These determinations will directly influence both the size and content of the lesson being designed. It must be emphasized that these parameters, once established, must not be departed from without a corresponding loss in design time and lesson effectiveness. Once the scope of the lesson has been determined it becomes necessary to formulate the construction process. This involves determining responsibility for various activities in the lesson's development cycle -- primarily the person responsible for formulating the lesson's contents, the author. There are currently three equally applicable options reflecting differing philosophies in this regard. These are 1.) AUTHOR/TEACHER, 2.) AUTHOR/EXPERT and 3.) MULTIPLE AUTHOR/MULTIPLE EXPERT. The first option suggests the person best suited to manage the lesson development process (including actual lesson coding) should be the same individual presenting the finished product in an actual classroom environment. Supporters of this option argue only those with considerable

classroom experience can identify the many special features of his/her discipline need be included in material presented in CAS form. Without disputing this assertion, those advocating the second school of thought are quick to point out few teachers have the time needed to first master a given authoring language and then do the coding and debugging the lesson will inevitably require. In many cases this constraint is the single most persuasive argument against the adoption of CAS methodology at many educational institutions. To counter this supporters of the AUTHOR/EXPERT philosophy offers the notion of the instructor as script writer while a data processing specialist with a practical working knowledge of the authoring language used be given the task of physically coding and debugging the lesson into final form. Given this division of labor the instructor is permitted maximum control over in desired results while minimizing the amount of direct involvement needed in actual programming related tasks. Whenever large CAS projects are contemplated there are often advantages gained in delegating responsibility for individual sections of lesson design to those having a particular expertise in a given area. This especially holds true in cases where courseware authoring is used to disseminate instruction in medicine or the hard sciences. Proponents of the MULTIPLE AUTHOR/MULTIPLE EXPERT approach advance the notion this rule be extended to all forms of lesson design, in all areas of interest. In this case groups of co-authors are formed to construct loosely organized sections of text integrated into a workable whole by a team of CAS specialists. The advantages of such an approach are a minimization of instructor effort while enhancing the degree of modularity, allowing common sections of text, to be "grafted" onto future lessons with corresponding reductions in design time. Any one of the three alternatives given here can be advanced with a reasonable certainty of success. Moreover, as script writer and program coder alike gain experience, the time required to produce workable sections of reusable CAS material will be decreased enormously. While each option can pose a set of advantages and disadvantages best evaluated through actual experience in accordance with one's own institutional capabilities and limitations, the emphasis here is the need to select a given process with clearly defined areas of responsibility and to stand by one's choice through the formation of the lesson without deviation from start to finish. One of the most critical decisions any lesson author can make will be the determination of his lesson's directional flow. No two common CAS lessons will be exactly the same. In point of fact each will follow one of several clearly defined directional patterns identified as either 1.) One-Way, 2.) Auto-Return, and 3.) Bi-Directional. The differences in these patterns come about due to the presence, or lack of, menus and branching routines at clearly defined breakpoints within the lesson. In the One-Way pattern there are no provisions for branching opportunities. In this case the student is presented with a lesson made up of a series of frames in the form of either informational displays or question & answer blocks. Lessons routinely proceeds from frame

to frame without the options of review or repetition until the lesson is completed. From the standpoint of the author this is the easiest pattern to adhere to. From the standpoint of the student, however, this focused direction is also the least imaginative and the one most to instill boredom with its lack of variation. The Auto-Return is much the opposite. In this pattern the student is automatically brought back to the lesson's main menu after the completion of a portion of the lesson. The student may now choose to redo a given section of material, or, if continuing from a previous study session, skip over one or more segments already mastered and continue at his own speed. While this variation can place flexibility at the hands of the user, it also requires a more structured lesson framework not every author may be comfortable with. The Bi-Directional pattern essentially combines the best of both options. While clearly the most difficult type of lesson to coordinate the benefits are commensurate with the time invested. Here students are free to proceed much as they wish. With multiple menus allowing for both review and skip options at different points throughout the lesson the student can branch to any logical start point within the lesson he wishes from any other point. In such a manner difficult sections of material can be repeated more than once, while those students can quickly master lesson content are no longer required to endure, for them, what would be endlessly repetitive examples and illustrations of simplistic concepts. The flow pattern selected will largely depend on factors such as conceptual complexity and over-all lesson duration. The higher the level of difficulty involved in a CAS lesson, the greater the need for a two-way flow pattern. Conversely, the shorter the lesson the lesser the need. Again selecting a proper directional pattern and maintaining it throughout the lesson avoids undue confusion on the part of the student and unnecessary complications for the designer. One of the more positive elements intrinsic to any CAS presentation is the ability to combine graphic representations of concepts generally difficult for the beginner to comprehend, in an effort to visually support what is being presented by the written word. While graphics, i.e. charts, graphs, tables, color and character size variations, etc., can collectively form a powerful enhancement to any traditional classroom methodology, the proper use of graphics in CAS designs is often misunderstood. Graphics improperly applied give much the same diminished effect as, by analogy, the presentation of advertising wherein the visual display, and not the product itself, is latter recalled. While the actual graphics used will naturally vary from lesson to lesson keeping in mind a few specifics can make for a more dynamic graphic presentation. With the very first frame the student should be presented with some type of graphic illustrative of the subject being covered. (An algebraic equation, a computer terminal, or a miniature Eifel Tower, for example, might be used with a lesson in mathematics, programming or French language instruction respectively). At the same time the opening frame of any lesson should display those colors -- no more than three -- selected for presentation purposes. Care should be taken in selecting color combinations that avoid eye strain over a period of time. Thereafter, in the following lesson frames a standard ratio between the amount of informational text and graphics included should be established. While this ratio can be varied slightly depending on the nature of the material being presented, maintaining this proper relationship produces a consistency helpful in creating a relaxing visual environment. Also in regards to visual consistency, authors should act to reserve a selected portion of their monitor screen in which text will always appear in the same location. Left and right hand screen sections can be used for specific functions. One example is the positioning of terminology in a left hand slice of screen with matching definitions on the right hand side. In

addition to this "windowing" effect, Headers presenting the name of the lesson prominently displayed can be psychologically reassuring to the student apt to feel lost half way into a lengthy presentation. Also, the author should likewise decide whether text included will be displayed through in both upper and lower case or capital letters exclusively. This initial standardization of physical appearance does much to reduce unnecessary confusion students might otherwise encounter. Never assume students can intuitively determine what responses are required at any given point. Even something as simple as entering a carriage return to continue the lesson at some given point can cause students with no prior exposure to computer aided instruction considerable frustration as they wait in vain for an instruction to proceed that never comes. Noting the modular approach to lesson construction alluded to previously, it is possible -- and indeed desirable -- for authors to include such directional aids whenever possible. The courseware authoring system is but one example of the advanced techniques now available to make the modern educator's role an easier one. In years to come courseware authoring will undoubtedly snare the responsibility of effectively dispensing quality education. To what extent CAS applications can be successfully utilized will depend largely on the ability of the instructor to maximize its effectiveness through intelligent applications. The above suggestions have been offered with these goals in mind.

THE KODALY FOLKSONG SYSTEM

Iza Gorofi and Robert Perinchief
University of Wisconsin-Whitewater

The Kodaly Folksong System is presented both for itself and as an application of information systems techniques. The system stores and retrieves songs by their musical and pedagogical characteristics, using both character and graphic displays.

The techniques used in developing business information systems are transportable to other disciplines. The information systems problems for the Kodaly Folksong System can be stated in terms which could apply to any business information system: develop an appropriate database and suitable user interfaces for the efficient storage and quick retrieval of information.

The Musical Problem

In current music education circles throughout the United States, there is widespread interest in an approach to music learning with roots in Hungary, identified with the late Hungarian composer, Zoltan Kodaly (KOH'-dye), and quite often labeled the "Kodaly Method" (1). More appropriately called the Kodaly approach, this philosophy-system assumes a commitment to folk song in the mother tongue, then folk song from many cultures, followed by utilization of fine composed music. The approach attempts to develop high music literacy among students, beginning at a very early age, even pre-school. The accomplishment of that literacy requires knowledge on the part of the teacher of a vast repertoire of folk music, so that song repertoire, and eventually instrumental literature, can be introduced in a structured manner for cognitive purposes. The music is classified by difficulty and complexity along a continuum superimposed on a taxonomy of musical elements complementary to the student's ability to comprehend and perform.

Therefore, a commitment to a Kodaly-influenced music curriculum in this country involves an ongoing search by the music educator for good folk song materials of all levels of interest and difficulty level, so that the Kodaly philosophy of beginning with the natural instrument, the human voice, can be applied to singing in early childhood that develops musical comprehension along with simple singing skills. The music educator searches constantly for songs which suit a purpose, namely, the physical ability of the child to sing the songs coupled with the intellectual ability of the child to comprehend the musical structure of the songs.

In the nearly two decades of the Kodaly "movement" in this country, numerous colleges and universities have begun to offer folk song research courses which develop the skill of researching song materials from this pragmatic viewpoint, instead of the traditional empirical viewpoint. Thus, teachers and students in such courses determine from one to two dozen categories under which to analyze the song materials they encounter, so that repertoire can be identified for one purpose early in the curriculum, a different purpose later. For example, it can be concluded that a frequent melodic pattern of two to four notes might be present in one song, making it a logical choice to place prior to or following a different song with a comparable (or even identical) melodic pattern in a slightly simpler or more difficult context. The very same song might have a recurring rhythmic pattern of two to four durations which makes it a logical choice to place at a different grade level, prior to or following a song with a similar or identical pattern in a context later in the overall curriculum.

If it is presumed, therefore, that most songs selected for the curriculum will have several different purposes, each perhaps at a slightly different place in the overall program it obviously becomes helpful to analyze researched literature for its multi-purpose application. It further becomes valuable to be able to analyze, store, and retrieve the various analyzed information based upon numerous musical items which have compared each song with other folk songs with common musical phenomena. At UW-Whitewater, complementary to a graduate course in folk song research based upon the Kodaly philosophy, an extensive one-page analysis form was developed in the early 1970's, which with revision has held for more than a decade as the basis for

storage. A complementary 3 x 5 card file has been developed cross-referencing the many subfacets of the analysis process. Other campuses have developed courses with similar storage and retrieval systems. With the advent of computer technology, it began to be apparent that the paper process was and would remain very limiting, and that a computer system would permit more widespread standardization and access for Kodaly-influenced teachers and students everywhere.

The problem became one of selectivity. From the nearly two dozen analysis categories used on the existing form in the UW-Whitewater Kodaly Folksong course, it had to be determined which were the most important for computer storage and retrieval systems. Could it also be possible to develop a visual realization of the single-melodic line on traditional music staff, with text below, so that the student might have access to the actual music score of the song, whatever analysis category unlocked that song with pertinent analytical information? Would it be possible to retrieve several songs with common identifiable rhythmic patterns, or melodic patterns, or patterns in form?

The multiple problem then became one of developing a computer system which would (1) store a visual realization of the song in musical score; (2) store title, book resource, and related non-musical information; (3) store analytical categories including melodic form, recurring melodic interval patterns, rhythmic form, recurring rhythmic patterns, scale/mode, range, tessitura (limited range within which most of the song lies), tonal ladder, melodic contour, musical cadences, meter, type of accompaniment, and others (2). If these individual problems could be resolved as part of the whole, the entire result would offer retrieval of an impressive array of information on many musical bases.

With the development of telecommunications systems permitting students statewide to learn from a scattered series of campus-based locations concurrently with the same instructor, utilizing audio and video interactive equipment, the problem becomes even more intriguing.

The User Interface

It should be the goal of all systems to provide an interface with the user that allows the user to "talk" to the computer in her/his own language. This is most especially true for users whose field is distant from information systems. The interface for this system can be thought of as consisting of three parts: menus, input screens, and reports.

Figure 1.

MAIN MENU

MAKE YOUR TRANSACTION SELECTION BY TYPING
IN THE NUMBER CORRESPONDING TO YOUR CHOICE

1. ADD A SONG
2. DISPLAY A SONG
3. RETRIEVE BY CHARACTERISTIC
4. EDIT SONG FILE
5. END

SELECTION: ___

In addition to the main menu there are several subsidiary menus. A feature of the system is that there are two categories of users, those who are allowed to enter information and receive reports, and those who in addition may check the entered information and delete songs from the system. The system prints only those portions of the menus for which the particular user is given access. A complete display of menus is not possible in this article due to lack of space.

Screens for add/update are shown in Figures 2-4.

Figure 2.

```

FOLK SONG ANALYSIS

TITLE : _____
SOURCE : _____
MELODIC FORM : _____
MELODIC INTERVALS : _____
RHYTHMIC FORM : _____
RHYTHMIC FEATURES : _____

CATEGORY : _____
MODE : _____
RANGE : _____

LADDER M, F, S, L, T, D R M F S L T D' R' M' F' S' L'

TESSITURA : _____
TONAL CENTER : _____
NEXT PAGE? (TYPE Y) : _____

TYPE "EXIT" TO RETURN TO MAIN MENU
  
```

Figure 3.

```

FOLK SONG ANALYSIS

CADENCES : _____
CONTOUR : _____
ORNAMENTATION : _____
VITALITY : _____
METER : _____
LANGUAGE : _____
SINGABILITY : _____
SPECIAL INTEREST : _____
ACCOMPANIMENT : _____
COMMENTS : _____

OPTIONS: 1 = ADD MUSIC          2 = ADD SONG
          3 = PRINT THIS SONG   4 = MAIN MENU
          5 = ADD ANOTHER SONG

ENTER YOUR CHOICE: _____
SONG WILL NOT BE ADDED UNLESS YOU TYPE "2"
  
```

Figure 4.

HALELUYOH

PITCH 13 12 11 10 9 8 7 6 5 4 3 2 1

COMMAND: INSERT XI A G # L S-TIE
 CHANGE XE B H # M TRIPLET T
 DELETE XX C I # N
 NEXT LINE XL D J # O P
 EDIT COMPLETE XC E J K # P Q
 GUILTY XQ F # R

EDIT TYPE: METER M EXAMPLE: 3/4
 KEY K BM (+ = MAJOR, - = MINOR)
 SYMBOL S
 TEXT T

ENTER COMMAND CODE: _____

A number of the input/edit and output screens required graphic displays. The music symbols were developed using REGIS graphics (3) on the GIGI terminal. Whereas the rest of the system was written in COBOL, the REGIS graphics subroutines were written in BASIC. VAX VM3 allows for linking COBOL programs with BASIC subprograms. Examples of output screens are shown in Figures 5-7.

Figure 5.

HILL AN' GULLY PAGE 14

BUT MY HORSE RAN STUM-BLE DOWN

HILL AN' GULLY
 NEXT PAGE (TYPE "Y"):

Figure 6.

```

FOLK SONG ANALYSIS

TITLE :HALELUYOH
SOURCE :EXPLORING MUSIC
MELODIC FORM :AA3+BBv
MELODIC INTERVALS :Si-F
RHYTHMIC FORM :AABB
RHYTHMIC PATTERNS :UVUV

CATEGORY : _____
MODE :HARMONIC MINOR
RANGE :M,M

LADDER M, F, S, L, T, D R M F S L T D' R' M' F' S' L'

TESSITURA :S,D
TONAL CENTER :L,
NEXT PAGE? (TYPE Y) : _____
  
```

Figure 7.

```

FOLK SONG ANALYSIS

CADENCES :V V V i
CONTOUR :AABB
ORNAMENTATION : _____
VITALITY : _____
METER :6/8
LANGUAGE :HEBREW
SINGABILITY : _____
SPECIAL INTEREST : _____
ACCOMPANIMENT : _____
COMMENTS : _____

OPTIONS: 1 = DISPLAY MUSIC          2 = ANOTHER SONG
          3 = PRINT THIS SONG   4 = MAIN MENU
          5 = RETRIEVAL MENU

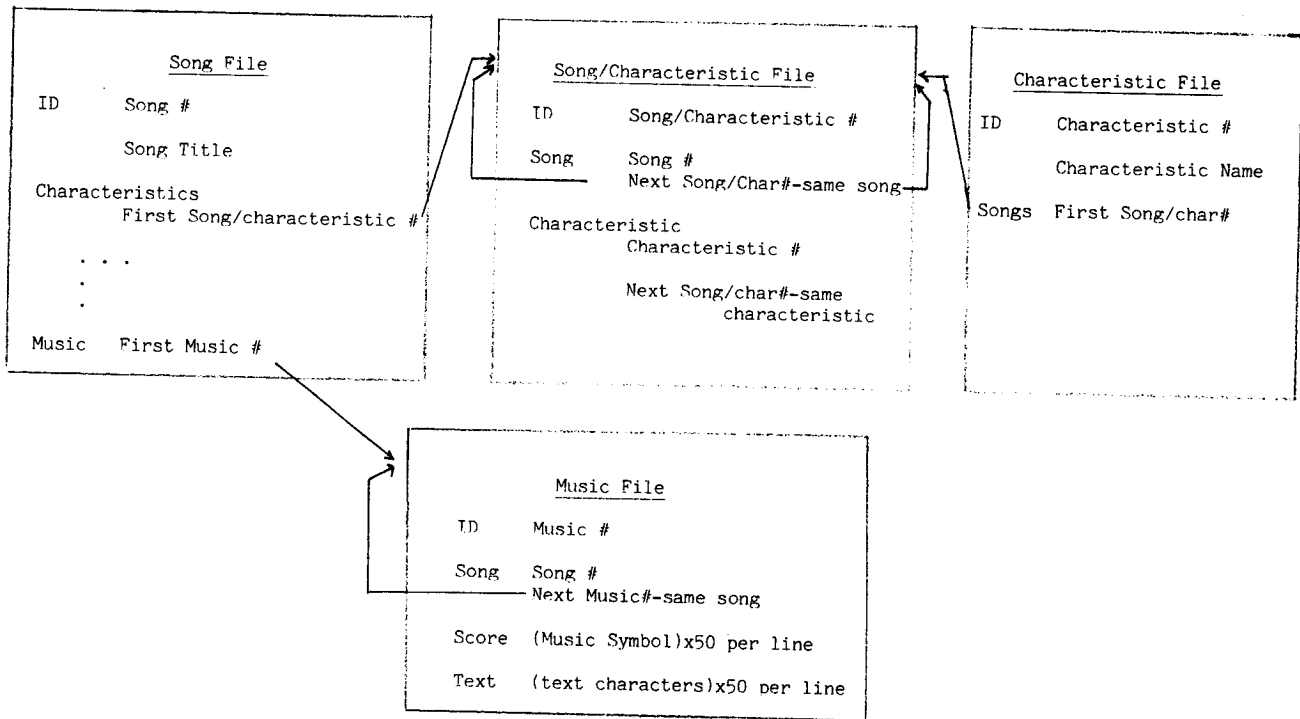
ENTER YOUR CHOICE: _____
  
```

Option 2 of the screen on Figure 7 allows the user to continue stepping through a linked-list of songs with the same characteristics. This is discussed further in the next section.

The Information Base

A major requirement for this system is quick retrieval of information by characteristic. "Quick" in this context means less than a second pause in writing to the screen, and in terms of the system design the COBOL verb "SORT" is not to be used. Instead an integrated database was designed to allow for the information to be stored in a presorted way using linked list concepts (4). Figure 8 shows the relationships between four of the eight files of the system.

Figure 8 INFORMATION BASE - INTERRELATED FILES SHOWING CATEGORIES, FIELDS AND INTERRELATIONSHIPS



To list all of the characteristics of each song the program first reads the appropriate song record to get the First Song/Characteristic #, the relative record number of the first of an indefinite number of limited records for that song in the Song/Characteristic file. The program reads that record to get both the Characteristic #, another relative record number, and the Next Song/Char#-same song. The program would read the Characteristic File to decode the Characteristic #. To get the next characteristic for the song the program reads the relative record given by the Next Song/Char#-same song and repeats the process listed above. The program loops until all the characteristics for that song are listed. The time taken by the file reads is much shorter than it takes for the printout.

A similar process gives all of the songs having a characteristic. This information base holds the many to many relationships of songs to characteristics with no maximum arbitrary number imposed, allowing for an indefinite expandability.

Credits

The system was developed by a student team as part of a one year Systems Analysis and Design (5) class project. The team consisted of Chris Cesarz, Cynthia Klement (team leader-first semester), Cheryl Nelson, Ralph Rinke, Mary Stolar, Elizabeth Zilen (team leader-second semester). They were supported by the personnel of the UW-W Computer Center, especially Al Krug, Atlee Svanoe and Lyle Hunter.

Conclusion

The techniques of business information systems can be profitably applied to other disciplines. We believe that experience gained by the systems people who do so may increase their sensitivity to be more creative in the business systems environment.

Iza Goroff, Management Computer Systems - Hyer 309,
UW-Whitewater, Whitewater, WI 53190
(414) 472-1468

Robert Perinchief, Music Department,
UW-Whitewater, Whitewater, WI 53190
(414) 472-1341

Bibliography

1. Choksy, L., The Kodaly Method, Prentice-Hall, Edgewood Cliffs, N. J., 1974.
2. Perinchief, R., Kodaly in Transparencies, Publication, Inc., Whitewater, WI, 1977.
3. GIGI TM/REGIS Handbook, Digital Equipment Corporation, 1981.
4. e.g. Augenstein, M. and A. Tenenbaum, Data Structures and PL/I Programming, Prentice-Hall, 1979, pp. 270-297.
5. Goroff, I., A Systems Analysis & Design Course Sequence, ACM/SIGCSE Bulletin, Vol. 14, pp. 123-127.

A TECHNIQUE FOR INDIVIDUALIZING MACHINE SCORED TESTS

RICHARD K. BREWER
Associate Professor
Computer Science Program
Xavier University
Cincinnati, Ohio

ABSTRACT: By combining a method for generating individual statistics tests and a method for creating open-ended machine scoreable statistics tests, machine scored individualized tests can be created. Students are asked to perform statistical manipulations on sets of numbers that include their own student numbers. This precludes most types of cheating. Numeric answers within prescribed ranges can be scored onto mark-sense forms by instructing the test takers to use a combination of the pre-printed answer positions to register a multiple digit answer, complete with decimal point.

When faced with overcrowded classrooms, large numbers of students, and established problems with cheating, instructors usually have a limited set of choices. Giving fewer tests but using multiple forms is one quite common solution. Another approach is machine scoring of objective tests with multiple forms, but the problems of producing multiple tests exert time pressures on a schedule already crowded with handling the details of scoring and recording scores for a large number of students. In addition, while it is simple and takes little time to produce a bad objective test, producing a good one takes more time than producing a good subjective test. The following approach was devised for a class in introductory business statistics taught by the author while a member of the Department of Operations Research and Information Systems at Eastern Michigan University under the conditions described. Introductory statistics offers a particularly manageable situation for devising individualized tests since so many of the beginning descriptive statistics can be tested using integer data that can be easily derived or generated. Students in the course were asked to compute such values as the standard deviation, the median, the mode, and the mean using a set of numbers which included each individual's own student number. At the institution for which this was devised, student numbers were assigned more or less arbitrarily, thus providing no reasonable clue in the final answer to a means of reconstructing a correct answer for anyone who obtains another class member's answer. Of course, anyone willing to risk the prolonged and therefore observable scrutinizing of another's work can derive the method, but most cheating accomplished in reasonably well supervised classrooms consists of obtaining completed answers. In order for individualized testing not to entail individualized key production at great cost, it is necessary to be able to generate individual keys automatically. Even with prepared keys, grading would be more time consuming than for standard tests without additional help. The author additionally chose to score the tests by machine. It is possible to obtain standard mark sense forms that permit ten different numeric responses to any question in the form of the digits 0-9 for any question. By combining several answer positions for a response to a given question, numeric answers can be coded directly, that is, not by selection from a set of pre-established choices. Thus the answers 0.000 through 9.999 (all 10,000 of them) can be coded using four answer positions. The number of digits allowed and the placement of the decimal can all be varied to suit the anticipated range of correct responses. Most such standard forms include an identification field in which the student codes her or his own student number in addition to the normal handwritten identification. This makes the student identification number available as data along with the answers, simplifying the scoring problem. Without the development of a large and sophisticated program that would allow correction from a special key, a computer program must be written to score each new

test given. That is, one program must be written that will score all responses to the same set of questions that differ only in the data to be used. If much use of this is anticipated, standard subroutines and functions in whatever language chosen (the author was using PL/I at the time) could be devised that would make the construction of such a program much easier in the future. The author used this approach only once, due to time limitations in developing the program. Difficulty came in an aspect that was not anticipated (though it certainly should have been), that of precision. The differing precisions of hand calculations, various electronic hand calculators, and computer calculations all had to be allowed for in each answer. This was finally accomplished by permitting a range of correct responses for each question. The program had to, of course, calculate fresh ranges for each separate response sheet. Answers can be pre-calculated for each student or calculated during the scoring process. The author chose the latter. The experiment itself was a success, qualified only by the problem the author experienced in completing the grading program in time. The tests were not graded until several weeks later after much effort was expended. The machine scoring principle for individualized tests using such an approach to individualization was proven effective and workable. Cheating was minimized as much through the realization of the students of the difficulty of cheating given the arrangements that had been made as through actual difficulty in cheating. A markedly lower average than anticipated for such a class at such a time was noted, however no student complained about the test being unfair or too difficult to take. It is doubtless true that in some cases the student was directed to the correct number of digits in the answer by a desire on the author's part not to waste response space, but this effect was probably minimal. To counter the evident distrust on the part of the students of machine scoring, question sheets were provided with both calculation space and a place for each answer. These were handed in with the machine scoreable answer sheets. Students were given back a machine produced form of the answers they gave keyed to the questions. If they had questions about the fidelity of reproduction of their own answers, they could come in for a conference at which time both the original question sheet and the mark sense sheet would be provided for comparison. This handled the few questions quite satisfactorily. Sample question:

5. Treating each digit in your student number as a separate number, combine these with the numbers below and compute the standard deviation for the entire set to three decimal places.

11,9,4,23,18

When you have calculated the answer, record the answer, digit by digit, in answer positions 21-24. Answer 21 should contain the leftmost or most significant digit in your answer and 24 the least.

Assume a decimal occurs right to the right of the most significant digit. If the student number was 156629, the calculations would be done with the series: 1,5,6,6,2,9,11,9,4,23,18. Hand calculation, a TI-30 calculator, and a VAX 11/780 computer using VAX BASIC all produced an answer of 6.401. A range of 1 or 2 one thousandths would be suitable to allow for a variation in precision, thus any answer between 6.399 and 6.403 would be considered correct. While this particular calculation did not show any differing effects due to precision differences, others might. In reality, some types of calculations could be expected to produce greater differences depending on whether the calculations were done with a computer, a time to determine reasonable ranges for each type of

hand-held calculator, or with pencil and paper. To allow for this without an unacceptable investment of calculation, a very broad range of answers that were accepted as correct was used. The author found that a range of two in either direction at the second least significant place solved the problem. The notion that a slight error down in the thousandths or hundredths place might go "unpunished" was quite acceptable to the author; principles of statistics were being taught, not neatness or beginning arithmetic. It was felt that such understanding could shine through the fog of trivial arithmetic error sufficiently for fair and adequate assessment.

CONDUCTING COMPUTER LITERACY WORKSHOPS
FOR FACULTY COLLEAGUES:
THE EXPERIENCES OF TWO WORKSHOP LEADERS

Kathleen Duffy

And

Paula Mitchell

Department of Computer and Information Science
Marshall University
Huntington, WV 25701

Computer and Information Science faculty are frequently called upon to bring computer literacy to their colleagues from other University departments. Traditional methods of instruction must be altered when the subject being taught is computer literacy and the students are faculty colleagues. This paper examines some of the differences in traditional classroom instruction and conducting computer literacy workshops for faculty colleagues. It focuses on the experiences of the authors in the conduct of such workshops, planning for the diverse abilities, interests, experiences and needs of faculty and suggestions for handling the inevitable problems that arise in the course of conducting these workshops.

Instruction of traditional college-age students brings with it a set of role expectations that govern the behaviors of the participants in the teaching-learning process. Students and instructors enter the milieu of undergraduate instruction with a shared understanding of the behaviors expected of each other. These behaviors often represent a typification of the characteristics of the ideal student or ideal teacher gleaned from the previous experiences of the participants. Students expect college instructors to be knowledgeable of the subject, prepared and organized, prompt in grading and returning tests and assignments, and tolerant and sympathetic regarding the students' lack of preparation or organization⁶. In turn, instructors expect students to be prompt in arriving at lectures, responsible in completing readings and other assignments, polite and properly deferential in addressing the instructor and reasonably alert and attentive during lectures.

These role expectations are rarely communicated directly but are formed from the gestalt of previous experiences. All instructors have experienced the role of student, usually for a significant portion of their lives. During their student experiences, they have synthesized their expectations of how teachers teach through direct contact with their own teachers and indirect contact with culturally recognized master teachers, e.g., Socrates, St. Thomas Aquinas, Don Scotias, and John Dewey. Further refinement and development as a teacher is derived from the process of teaching itself.

In the course of conducting computer literacy workshops for faculty colleagues, the authors discovered that the role expectations governing the interactions between students and their instructors often break down when the students are colleagues. Behaviors that would be intolerable from students are often tolerated or at least overlooked in one's colleagues. Tardiness, lack of attention, jumping ahead in the material or skipping tasks altogether are behaviors exhibited by at least one participant in each workshop.

A factor contributing to the difference between teaching colleagues and teaching traditional students may be that colleagues have specific needs and goals related to their established careers. They are accomplished adults and recognized experts in their own fields. Maintaining their self-esteem is as important as mastering the use of the computer. In contrast, students more readily accept broad based instruction because their needs are not yet as specific or narrowed.

Non-computer science faculty participate in computer literacy workshops for a variety of reasons and often with vastly different expectations about what will be accomplished in the workshop. Many faculty are enthusiastic and motivated to incorporate the technological advance of microcomputers into their teaching or to help them with research and writing. A few come with an attitude of stoic acceptance of the inevitable, "It's (the microcomputer) here to stay. I might as well learn to use it." Others attend because of perceived pressure to incorporate microcomputers into their curriculum or because they fear being left behind by more knowledgeable colleagues and students. Still others participate because they will obtain Continuing Education Credits (CEU's) or because doing so contributes to their professional development in a real sense or by "enhancing their vita."

When faculty members attend computer literacy workshops due to pressure from others or fear of being left behind, resistance to learning how to use the computer may be encountered¹. Experienced faculty who are comfortable with their established teaching techniques may react negatively to the idea of incorporating the computer into their curriculum, and, to the workshop introducing them to the use of computer technologies. To counteract this resistance, workshop leaders must be enthusiastic, positive and most of all -- patient. Workshop materials that directly benefit participants help overcome these feelings of "computerphobia."

Irvine² and Lovell³ note that feelings of "computerphobia" or "technophobia" are not uncommon among novice computer users. The present authors have also found that frustration is a typical response among workshop participants who believe that "user friendly" personal computers "do everything for you" or make it possible for everyone to become an instant computer expert.

Some of our workshop participants have commented with surprise that "the system commands for the IBM-PC and APPLE IIe are so complicated" or disappointment that we could not teach them BASIC, WordStar and Visicalc in a 1-3 day workshop. In the course of conducting computer awareness workshops over the last several years, we have found it necessary to emphasize to participants that practice and process time are needed to master the use of microcomputers and their software -- even though much skill and knowledge can be obtained in a relatively short time period.

The knowledge, ability and previous computer experience of workshop participants usually varies tremendously as well⁴. Even in a workshop directed toward introducing the novice to the use of microcomputers, workshop leaders find that participants range from the complete non-user to the very experienced user with knowledge of a programming language or experience with word-processing, text-editing, spread-sheet or statistical software⁵.

The authors have found it common among workshop participants who have had some previous experience to question the purpose of the workshop or to attempt to redirect its content to one of more immediate personal utility. This experience underscores the need to assess the computing knowledge and interests of workshop participants before the workshop -- to separate the experienced faculty from the true novice and to plan for the different needs of those with a modicum of previous experience and knowledge.

We have learned that teaching colleagues often requires that instruction be more individualized than it is when teaching the same material in undergraduate CIS classes. The suggestion has been made by several workshop participants that we provide one-on-one instruction. While recognizing the feelings that spark such requests, we also realize how unrealistic it would be to conduct one-on-one workshops. Workshops are typically scheduled for 1, 2, or even 4-hour time blocks over several days or weeks. One-on-one instruction would drastically limit the number of workshop participants as well as limiting the workshops we would be able to offer to our colleagues in our spare time.

Because workshop participants typically represent various departments within the University, topics must necessarily be limited to those of general interest and wide applicability, e.g., word-processing or spread-sheet software. However, on several occasions our workshops have been interrupted by participants who wanted to know how their particular department would make use of the microcomputer. As we have grown in experience we have learned how to field these unwanted interruptions just as we do in undergraduate CIS classes -- by deferring the answer to a later time or suggesting a little personal research to the questioner. On occasion, we have pointed out that what may be of particular concern to someone in the field of Economics or Sociology may be of little interest or relevance to the faculty in Accounting, Finance, Nursing or Education.

We have been surprised that topics of seemingly universal application such as word-processing or spread-sheet software are so often received with mixed interest. Some faculty view word-processing software as a tool that will provide them with added control and efficiency in their writing and report preparation. Yet other faculty resist learning word-processing, viewing it as primarily in the domain of clerical staff. Many faculty show an enthusiasm for learning to program in BASIC or other high-level programming languages while others see greater utility in using commercially prepared software. Such diverse interests cannot always be addressed in a general computer literacy workshop designed to raise the participants' computer awareness.

What then are the rewards and compensations of conducting Computer literacy workshops for one's colleagues? On occasion, both authors have received stipends for conducting workshops but this task is usually performed as a service to the University or

community at large. Performing this service has raised our visibility among colleagues and contributed to our professional development and recognition as a resource for continuing professional development to our peers.

The greatest satisfaction derives from accomplishing the purpose of a computer awareness workshop for even a few of the participants. Colleagues who comment "I learned so much in your workshop only to see that I have so much more to learn" and then make use of what they have learned with their students are both the reward and the incentive to conduct future workshops.

REFERENCES

- ¹Ewert, A., Employee Resistance to Computer Technology. Journal of Physical Education, Recreation and Dance, 1984, 55(4), 34-36.
- ²Irvine, P., Introducing Faculty to Microcomputers. Health Education, 1983, 14(6), 13-14.
- ³Lovell, P., Staff Development for Computer Literacy. Educational Technology, 1983, 23(3), 18-19.
- ⁴McMeen, G. R., Developing Computer Literacy Workshops for College Faculty. Educational Technology, 1984, 24(4), 25-29.
- ⁵Secrist, I. S., How to Keep Users Happy and Keep Your Sanity. Computer Decisions, 1980, 12(2).
- ⁶Waggoner, M., The New Technologies Versus The Lecture Tradition in Higher Education: Is Change Possible? Educational Technology, 1984, 24(3), 7-12.

THE FUTURE ROLE OF MICROCOMPUTERS IN AUDITING:
AN EXPERT SYSTEM EXPLORED
by: Debbie Dare
Student at Glassboro State College, Glassboro, NJ

ABSTRACT

This article shows how Expert Systems will be used in the auditing environment. This is done first by extracting data from the source database into a microcomputer's database. The Expert System will use concepts from artificial intelligence to provide an opinion on reliability and accuracy of financial statements. Additionally, Computer Business Graphics will provide management with an understandable representation of the company's financial position.

The winds were gusting now up to 15 knots as Amy Durham sailed along the coast of Santa Barbara, California, in her 32-foot O'Day. In the bright afternoon sun she began to write her weekly log report of progress on the new system used by her firm, Durham and Cooke, Inc. This was to be submitted to the senior vice-president the following Tuesday, June 15, 1999. Her main objective in writing this review was to analyze the new Decision Support System and the development of a new database. She also was to explain the use of a new sophisticated system known as Expert Systems and how her microcomputer had the capacity to hold this system. A new tool for management was Computer Business Graphics which aided management in future projections of the company. The sun began to fall below the horizon as Amy began her examination of the company's new strategy.

Amy knew that planning a Decision Support System would be a long and tedious task. Earlier Durham and Cooke, Inc. had developed a simplified Decision Support System (DSS) as a starting point for the company. Using a technique known as Iterative Design, the company had developed a Specific Decision Support System (SDSS) applicable to its individual needs. This process seemed most appropriate for the company's system because of the need for flexibility. With the changes in our society the firm had these changes to meet.

Keeping this in mind, two objectives of the new system were met. First, no one at Durham and Cooke, Inc. could accurately predict what would be future trends; therefore the company would continually evolve and grow with the changing society. Secondly, the system would never be final. It must always change to meet the needs of the user and the environment and to meet needs of future problems. The Iterative Design process requires the active participation and communication of the builder and the user during the operation and evaluation of the system.

Through continually modifying and developing the system the new Specific Decision Support System evolved. An example of the Iterative Design process is that of building a house. Steps are taken to draw plans and begin building. The process is continually changing and the house is always developing through the years. When the final structure is built only refinements need to be done.

The winds began to shift, and Amy had to begin tacking from east to west along the coastline. She then began her analysis of the Data Extraction concept in Durham and Cooke's Data Base

System.

This design is a technique for "interfacing a variety of source databases with a DDS database".* Source data includes both internal and external data. Internal data includes files for inventory, accounts receivable, and parts and supplies. External data would contain the outside evaluations of the company from sources such as S & P 500, Value Line, and Dow Jones. This source database may cover "several logical files using various formats which may be stored on separate computer systems."** Through using the Data Extraction System the user can "extract" from the source database information that would pertain to users objectives. As an example, the auditor would only extract client records which are to be audited. This extracted information is downloaded by means of telecommunications from the mainframe computer to the user's microcomputer which stores the newly formed database on hard disk. In other words, the user creates her own individual database by accessing the source database.

Extraction operations allow the user to interface directly with the source database in four unique operations. The first operation is data description which describes files in the source database. The second is the subsetting operation which enables the user to select fields from the source database using arithmetic and logical criteria.** Aggregation is the third operation which allows fields or records to be summed, joined, counted, or combined in any manner. The final operation the user can apply to creating a database is the data presentation. The purpose of the user may not be to include the aggregating and subsetting operations in the Extracted Data Base, therefore the operations may be displayed on the CRT. The contents of this Extracted Data Base are the past and current year's records, industry averages, and historical trends of the company. These are important tools used by the auditor in performing compliance and substantive tests.

This type of Extraction System has been advantageous for Durham and Cooke, Inc. for a few reasons. The first advantage is that when using subsetting and aggregation operations, the computation and access time is cut into one operation rather than repeatedly extracting from the source database. Secondly, security within the company has tightened because the assigned auditor has access to the source database which requires authorization. Once the user creates her own database she no longer uses the company's source database. This decreases the risk of (1) exploitation of client's records and files and (2) in some cases the negligence or intentional misuse of records and files. Confidence is the direct responsibility of the auditor. Finally, the source database files can be organized more efficiently for easy access. This allows filing updates, processing, outputs, or protection of

*Sprague, Ralph H. and Eric D. Carlson. Building Effective Decision Support Systems. Englewood Cliffs, NJ: Prentice-Hall, 1982, p. 244.

**Sprague, p. 245.

***Ibid.

the system, without having to add more data to support the DDS. Thus, processing costs are reduced.

The wind had slowed to six knots allowing Amy to dock her sailboat "Euphoria". As she floated in to the slip she thought about the contrast of audits now as with "back in the ole' days". Today they are much faster and accurate. No more does she have that 9-5 job, five days a week. After tying up her sailboat she had a few more thoughts she wanted to write down before returning home.

The conventional way of performing an audit, as done in the 1980's, is time-consuming, and costly when considering today's techniques. Since the discovery of Expert systems the concept of artificial intelligence can be applied to accounting. Through developing software for micro's, audits can now be performed procedurally. Inference mechanisms are added to enable the system to use this knowledge of performing an audit in a problem-solving environment. The knowledge needed to build this system comes from users of existing systems, experts in the field, past historical data, textbooks, and literature.

Broadly speaking, in constructing a system for artificial intelligence the software must perform tasks normally accomplished by human intelligence. EDP Auditors cannot state each reasoning process while performing an audit. However, there are certain "rules of thumb" which they apply during an audit. Collecting these rules allows the Expert System to analyze more complex problems. A key feature of the system is the great flexibility it holds. Rules and facts can be easily added, deleted or modified by the user to create her own personalized knowledge base.

Using our Expert System, performing an audit is done with ease. The Expert System is programmed to do a full random sample of the client's records before the Extracted Data Base is created. This random sample database is downloaded, using telecommunications facilities to the microcomputer. When the downloading has been completed, a question appears on the screen, "Do you wish to continue?" At this point the auditor types "Yes". When the menu appears, the auditor keys in the sample size, the confidence level and the precision range. When discrepancies appear during the process an automated error listing is printed for the auditor. The error listing may include internal control problems, fraud, and missing invoices. From this point, the auditor utilizes the teleconferencing system in the office. This is used to contact a team of paraprofessionals at the plant to find the discrepancy physically. An example of this is physically counting inventory or finding the hard copy of the invoice. The size of the team depends directly upon the size of the client's plant. At a large plant a team may consist of 25 or more whereas at a small plant only one or two are required. While the team is looking for the imbalance, the auditor continues to interact with the microcomputer. Questions will be asked in an "IF-THEN" format to determine if the discrepancy is in the system. When the auditor asks the computer "Why?" a listing of the rules of thumb used to reach these discrepancies are immediately displayed on the CRT.

A subset of the Expert System is the Expert Information System. One type of Expert Information System used by Durham and Cooke, Inc. is TAX-ADVISOR. Through using this the company can make tax planning recommendations to enhance the wealth of clients based on the client's current year operations. TAXADVISOR was developed by Michaelsen in 1982 under the supervision of Michie at the University of Illinois. This system actually performs the consultant's role by using already programmed rules of thumb to determine planning alternatives available to the

client. Only the "human tax consultant" interacts with the system, not the client. TAXADVISOR will ask the consultant "Yes/No" type questions pertaining to the client. When they are answered by the tax consultant, the system displays its recommendations for the client on the CRT. If the user does not understand how a decision is reached, she simply asks the system "Why". The system will display its reasoning behind the decision using the rules of thumb contained in the system. TAXADVISOR is an important tool for the success of Durham and Cooke, Inc.

Amy tied up her boat and was headed back to Malibu for the dinner party she was hosting for the office. As she drove back she began to make a few final mental notes on her analysis of the new system.

Once the system has completed the audit internally, it automatically generates reports and an analysis of the company. These include an opinion of the company, its pertinent financial statements, and an analytical analysis in the form of reports and graphs for the client. Opinions and financial statements have not changed over the years; however, the new graphical reports are new to our company's performance of an audit. Through working with management Amy has seen the need for an alternative to pages of numbers in order for management to understand its current financial position. Two attitudes have developed by management. First tabular reports such as the balance sheet are poorly designed, and they contain information not needed to operate efficiently and effectively. Second, any relevant information needed should be condensed to one or two pages. Management no longer has these two attitudes because of Computer Business Graphics (CRG). Computer Business Graphics is the "science and art of transferring information through the sense of sight."*

When the auditor completes an audit, the opinion and financial statements are transferred to the junior vice president of Durham and Cooke, Inc. The junior vice president then inputs the final account balances in the microcomputer. This information is transferred into bar charts and time series charts in color and two and three dimensional graphs. It has been the experience of Durham and Cook, Inc. that graphics are an effective tool for management's use in evaluating previous year operations and forecasting future operations. The first set of graphs is that of financial ratios. A return on sales ratio illustrates that budgeted return on sales was greater than actual return on sales (Figure 1). As the ratio is now broken down further, we find that sales were acceptable and net income was actually below what management had projected. At this point management would look into the components of net income as a solution to this problem. A set of ratios can also be computed using past year ratios with current year ratios comparing them graphically (Figure 2). Then component charts are graphed. As an example, the Income Statement is graphed to show the relationships among its components (Figure 3). Variance charts are perhaps the most illustrative of Computer Business Graphics. Management can readily see a region that has a negative variance (Figure 4). This looks more sensible to the eye than a page full of numbers. The junior vice president will put together a portfolio of graphs for management to use as a tool for improving operations and improving strategic marketing position.

Pulling up to the driveway, Amy Durham thought how the accounting job had changed so much since back in the 1980's. Today, she thought, we are more useful to management and more efficient in

*Jarett, Irwin M. Computer Graphics and Reporting Financial Data. New York: John Wiley and Sons, Inc., 1983, p. 3.

our work. It has taken many years to develop this system, requiring a lot of hard work; therefore, it has become a viable and worthwhile system.

BIBLIOGRAPHY

Davis, Daniel W. Assistant Professor, Glassboro State College, NJ, Consultation, December, 1984.

Hannan, James. A Practical Guide to EDP Auditing. Pennsauken, NJ: Auerbach, 1982.

Hansen, James V. and Messier, Wm. F., Jr. "Expert Systems for Decision Support in EDP Auditing." International Journal of Computer and Information Sciences, 11N5 (October 1982), pp. 357-379.

Hayes-Roth, Frederick, Waterman, Donald A., and Lenat, Douglas B. Building Expert Systems. Reading, Mass.: Addison-Wesley, 1983.

Jancura, Elise G., and Roos, Robert. Establishing Controls and Auditing the Computerized Accounting System. New York: Van Nostrand Reinhold, Co., 1981.

Jarett, Irwin, M. Computer Graphics and Reporting Financial Data. New York: John Wiley and Sons, Inc., 1983, p. 3.

Lenat, Douglas A. "Computer Software for Intelligent Systems." Scientific America, September 1984, pp. 204-213.

Lins, William C. "The Microcomputer in Accounting Education." Symposium at Richard J. Hughes Justice Complex Conference Center, Trenton, NJ, November 1984.

Michaelsen, Robert and Michie, Donald. "Expert Systems in Business." Datamation 29 (November 1983), pp. 240-246.

Sprague, Ralph H. and Carlson, Eric D. Building Effective Decision Support Systems. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1982, pp. 244-245.

Weber, Ron. EDP Auditing Conceptual Foundations and Practice. New York: McGraw-Hill, 1982.

Figure 1

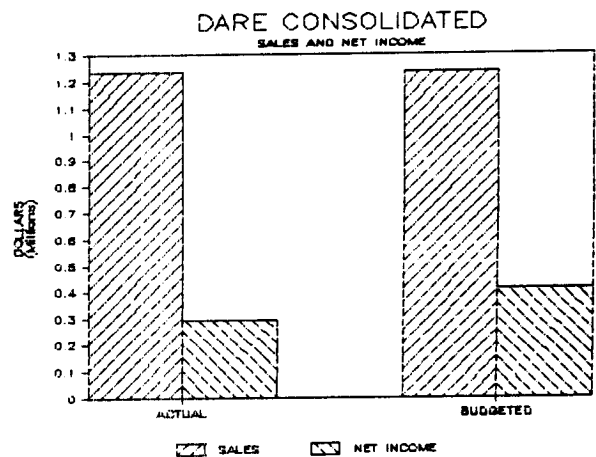
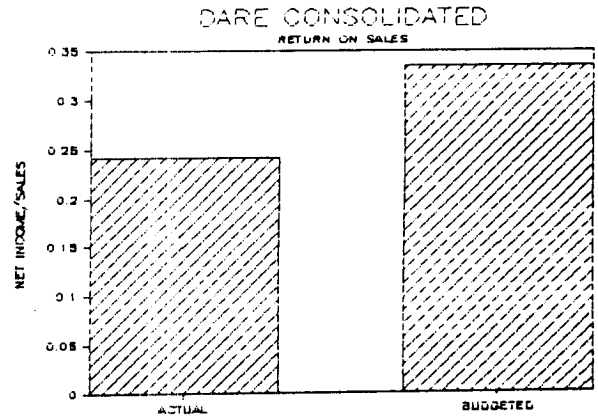


Figure 3

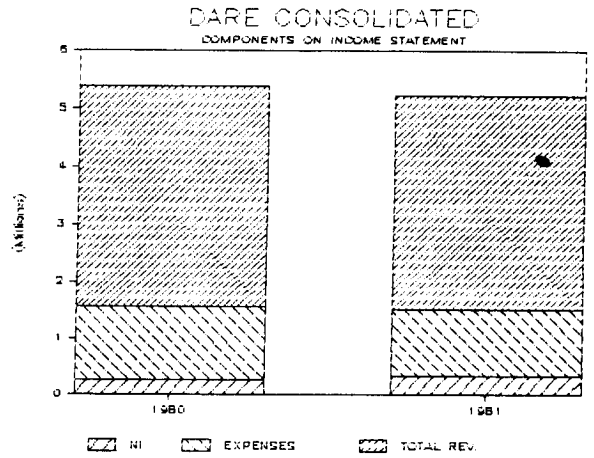
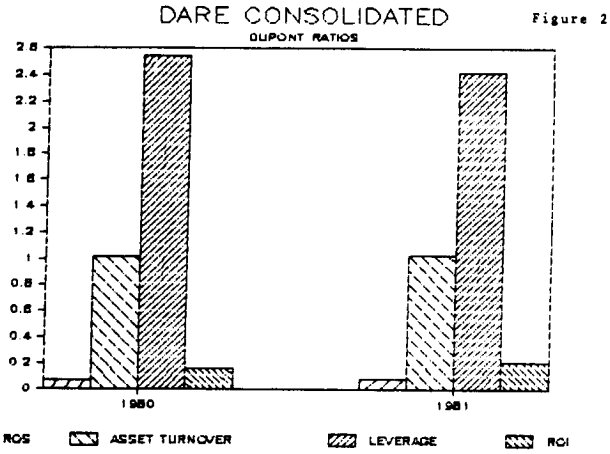
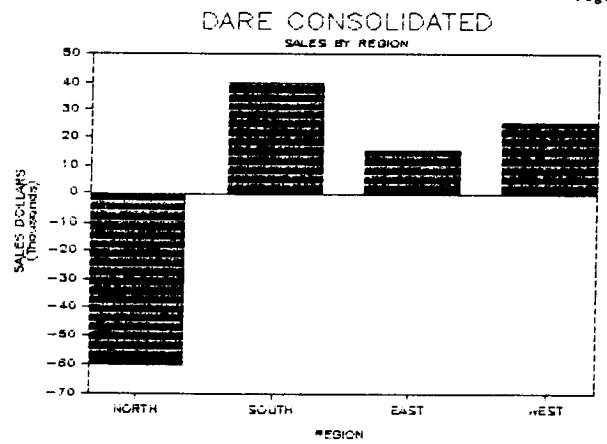
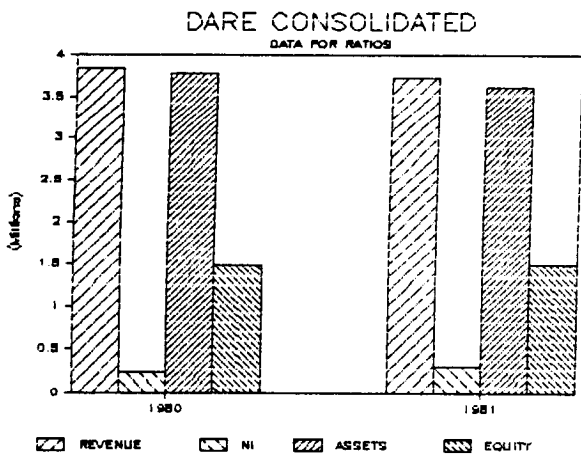


Figure 4



THE PSYCHOLOGICAL EFFECTS OF AUTOMATION ON OFFICE WORKERS

Karen A. Lensler
James Madison University
Harrisonburg, VA 22807

This paper discusses the impact of automated office technology on the office worker. The individual's perception of automation as threatening and dehumanizing leads to psychological feelings of alienation. These psychological effects of office automation on workers is an important issue which needs to be addressed for the success of the working environment.

For millions of people, work will never be the same again. Jobs that used to require filing and typing now require punching computer terminal keys and reading words displayed on screens. Today, offices are more and more automated. Devices such as the electric typewriter and the telephone have become so commonplace that virtually all office workers take them for granted and few view them as threatening. However, during the last few decades, automated systems in offices have exploded to the point where many now have photocopiers, word processing systems, and desk-size computer terminals. Although there is no question that these improvements make many tasks easier, they don't necessarily make jobs easier or more pleasant. Ultimately, the greatest impact of automated office technology will be on the office worker. The problem is that workers have their own perception of what office automation is like. Today's white-collar worker feels threatened by modern developments in office automation; included in this category are technology, organization, and implementation. This perception leads to psychological feelings of alienation in the individual.

TECHNOLOGY

Office workers fear office automation because of their perception of the new office technology. They view technology in various ways. Firstly, many see computers as cold, metallic, and impersonal.(a)* They see the computer as an inanimate device because of the objectivity and uniformity with which it treats all of its users. In addition, employees picture the automated office as one of sterile, factory-like efficiency in which human workers are little more than drones serving their computer masters.(b)* Office workers dislike some of the dehumanizing aspects of automation such as its unrelenting pace, its intolerance of human error, and its requirement that operators communicate in a coded language. This language jargon bears a mystique which tends to frighten, or at least discourage many people.(c)*

Secondly, the reputation of data processing centers discourages the employee from accepting the concept of office technology. This reputation has always been one of elitism. Data processing personnel have the notion that they are inherently superior to and have better judgement than the computer customers; they feel that they are the "masters of the computers." In addition, poor experiences in using data processing services have contributed to employees' distrust in the new office technologies being implemented into their own offices. Even after new procedures are established and users problems ironed out, users still hesitate to use their computers because of its past reputation.(d)*

(a)* Mary L. Schraml, "The Psychological Impact of Automation on Library and Office Workers," Special Libraries, April 1981, pp. 149-155.

(b)* David Steinbrecher, "Automation Outlook," Office Administration and Automation, May 1984, p. 8.

(c)* Schraml, p. 149.

(d)* "Weighing Word Processing Work Promotes Productivity," Modern Office Procedures, Oct. 1981, p. 54-57.

In addition, the fear of misuse of information stored in computer files has caused considerable concern for office employees. Workers believe that computers have the potential for increasing the control of organizations over its members and for invading the privacy of its members. Workers have seen that computer fraud is common whenever information is exchanged with Computer Based Information Systems; consequently, "lockpickers" are abundant. Lockpickers are computer hackers who find it ridiculously easy to browse through other's files.(e)* Workers are concerned with the usability of workstations. They see terminals sitting on many desks in the office that are used for data entry and retrieval, time sharing, and word processing. They want to know whether office workers who are not direct users of computers use these terminals.(f)* Therefore, the security of office systems is questionable and it appears management ignores this problem.

Finally, since workers fear the technologies of office automation, money is wasted. Employees are averse to these systems and refuse to use them. Not only is this hesitation evident at the lower level, but also at the executive level. Many executives often complain, "Why do I have this terminal on my desk?"(g)*

ORGANIZATION

Organization policy changes is another reason office workers fear office automation. When technological change is mentioned, the first thing many people think of is unemployment.(h)* Unquestionably, workers are primarily concerned with the organization's unemployment policy. Workers believe that automation means the displacement or replacement of people, reduced opportunities for advancement, and feelings of human obsolescence. This belief is true because as more work is absorbed by the new automation systems, organizations need relatively fewer employees to do a given volume of work. For example, last year, Equitable Insurance fired over 500 middle-aged employees, declaring that its old policy of lifetime employment has become inconsistent in recent years and that "times have changed."(i)* The new technology introduced to the insurance company made the middle-aged insurance workers' skills obsolete. Workers see that the problems of displaced office workers are particularly serious for older workers because once unemployed, they are less likely than younger workers to be re-employed. Therefore, in order to keep up with the times and remain viable, employees are forced to upgrade old skills or develop new ones. However, this is not always possible due to constraints in time, lack of management support, or an inability to learn new tasks.(j)* In addition,

(e)* Alexia Martin, "The Human Connection: A Strategy for Making Automation Work," Administrative Management, Feb 1982, pp. 33-35.

(f)* Martin, p. 33.

(g)* Lad Kuzela, "Confronting Computer Phobia," Industry Week, 19 Sept. 1983, pp. 91-92.

(h)* Schraml, p. 149.

(i)* John Thackray, "White-Collar Blues," Management Today, March 1980, pp. 94-102.

(j)* Schraml, p. 150.

although computers have moved into a diversity of applications, organizational members, especially supervisors and lower-level managers, are not always prepared to cope fully. Managers' computer preparation often lags behind installation and their computer knowledge is usually incomplete.(k)*

Workers are also uneasy about structural changes in their jobs. If jobs are not eliminated entirely, they are broken down into less skilled, lower-paid positions.(l)* The de-skilling of jobs reduces a worker's ability to make decisions and as a result, there is a loss of job skills and know-how. The "office of the future" becomes a "white-collar sweatshop"; the modern office, a factory.(m)* In addition, they believe job de-skilling undermines the image of white-collar workers because it allows no room for advancement. Options for job advancement existed in the past; however, nowadays, a person can be stuck.(n)* Until management recognizes that changes in technology are frequently interpreted as closing off promotional opportunities, increasing job releases and relocations, and de-skilling tasks, workers will be frightened of the computer.(o)* This perception is held by many workers, especially those employed by the large U.S. insurance companies, who believe organizational policy as a result of office automation has turned from one that values experienced employees to one that treats them as superfluous.

IMPLEMENTATION

Office employees also fear automation because of the way computers are implemented into their work environment. Workers believe that the computer technology under which they work was designed without regard to the "human" factor in the system.(p)* The people who design computer systems are typically computer scientists and systems analysts who have no concept of the human element in office automation. Consequently, a contributing factor to the distrust of automation has been the systems designer's lack of consultation with the operators as well as the users who are directly affected by the implementation.(q)* When there is little user involvement, workers see office automation leading to a dehumanization of the work force activity that is similar to manufacturing assembly lines. As a result, their work content becomes simplified and boring. As such, the machinery becomes a source of misery rather than a helpful tool.(r)* However, when users do participate in the implementation, it is often difficult for users to translate their requirements into terms that the designers understand. The fault lies equally with the systems designers who know their systems and technology but do not understand the business problems.(s)* In addition, systems analysts do not understand users' anxieties and fears of mastering new technology. Therefore, automating a firm unfortunately amounts to little more than an unvarnished campaign of employee intimidation.(t)*

(k)* Schraml, p. 150.

(l)* A. Cakir et al., Visual Display Terminals (Chichester: John Wiley and Sons, 1980).

(m)* Steinbrecher, p. 8.

(n)* John J. Connell, "White Paper: First in a Series," Modern Office Procedures, March 1982, pp. 51-64.

(o)* Jim McNitt, "Making Computers and People Compatible," Nation's Business, March 1984, pp. 50-52.

(p)* U.S. Department of Health and Human Services, An Investigation of Health Complaints and Job Stress in Video Display Operations (Cincinnati, Ohio: 1979).

(q)* Schraml, p. 149.

(r)* Martin, pp. 33-35.

(s)* Martin, pp. 33-35.

(t)* McNitt, pp. 50-52.

For example, workers get nervous when asked to itemize their tasks (costs, number of hours). As a result, staff morale is affected. No one likes to justify his job. As such, the approach to resistance is made before the computer arrives.(u)*

Secondly, workers have the perception that jobs are designed around machines. They believe jobs should not be structured around machines. Rather, machine systems should be designed so that jobs can be carried out more efficiently.(v)* Workers see that people and work must adjust to the computer rather than vice versa. Therefore, when this occurs, workload requirements tend to be set more by the capacity of the machine than of its human operators. The fact that a computer can now process millions of bits of information per minute does not mean that human capabilities have increased proportionately.(w)*

Lastly, employees believe that automated systems do not fit with the job because many managers use the bottoms-up approach to computerizing a firm.(x)* It starts when one department, like accounting, gets a computer and they become to be known as the "computer department." Consequently, the top brass hears about this and buys personal computers for the whole company without seeing whether these computers will be efficient in all of the departments.(y)* Top planners are commonly seduced by technology to the point of not being able to develop a strategy for its practical application.(z)* As a result, the modern office becomes task-oriented rather than goal-oriented. Therefore, employees, the organization's most valuable resource, are not involved in making choices for their future.(a)*

PSYCHOLOGICAL FEELINGS OF ALIENATION

Since office workers feel threatened by the developments in the modern office, they suffer from feelings of alienation. There are several related factors which contribute to the alienation people feel towards machines. Firstly, workers feel a lack of control over the machines with which they work. Employees believe that office technologies rob them of their skills and decrease their control over their work.(b)* For example, operators are monitored closely by the computer systems which provides supervisors with up-to-the-minute performance reports on the rate of production and error levels.(c)* This use of the computer to control individual levels of performance can constitute a significant amount of mental stress.(d)* In addition, remote supervision causes many employees to limit their own risk-taking behavior (spotting an error and correcting it or developing a more effective approach to tasks). Workers feel that they are constantly watched by a computer rather than by a human supervisor who could offer feedback and assistance in improving the quality of work.(e)* As a result, workers feel constant pressure for performance.

(u)* McNitt, pp. 50-52.

(v)* Connell, pp. 51-64.

(w)* U.S. Department of Health and Human Services, Potential Health Hazards of Video Display Terminals (Cincinnati, Ohio: June 1981).

(x)* McNitt, pp. 50-52.

(y)* McNitt, pp. 50-52.

(z)* Martin, pp. 33-35.

(a)* Martin, pp. 42-44.

(b)* "An Investigation of Health Complaints".

(c)* "An Investigation of Health Complaints".

(d)* Cakir.

(e)* Schraml, pp. 149-155.

Secondly, office automation decreases worker's interaction with other people.(f)* When the computer transforms a job that once required extensive interaction with fellow workers into solitary days at a keyboard, even the most personable employee may grow sullen and resentful.(g)* In addition, people feel isolated and chained to their terminals, not engaging in interpersonal relations the way they had before. This spending a large portion of one's day communicating through a terminal rather than face to face with others diminishes worker's communication skills (observed at video arcades and among personal computer buffs). This loss of skills causes teamwork, participative management, and other group-oriented practices to suffer greatly. Soon, departments lose contact with each other and the communication process between co-workers deteriorates. Furthermore, there is a lack of recognition of individual excellence when workers rarely communicate with others beyond their immediate supervisor. It is important for top management personnel to be aware of worker's good performance.(h)*

Finally, office automation causes workers to feel that their work is not important -- an exercise without meaning or purpose. There are several reasons for this belief. Workers feel their jobs are so specialized, reduced in scope, and monotonous that anyone could do their job.(i)* All that is needed are people who can read well enough to push the correct buttons on the machine.(j)* The work is so divided up that no one has the total picture. In addition, employees feel machines are more intelligent than humans. This overrated power, given to the computer, as an indispensable tool creates a feeling of inadequacy on the part of the workers. As a result, they feel their intelligence is no longer appreciated; the computer has co-opted their judgement.(k)* Furthermore, jobs do not give workers the richness of life necessary for personal growth and development. For example, a file clerk who becomes the operator of an electric information retrieval system gains power over stored data and should feel important as a result. However, this same clerk feels deprived of the interactions and satisfactions her previous manual job gave her.(l)*

CONCLUSION

The psychological effects of office automation on workers is an important issue to which managers need to address. Unless managers consider the attitudes individuals have towards the technology of computers, the organizational policy changes, and the implementation of automated systems into their work environment, office automation can have serious consequences. The manager who is unaware of or is insensitive to the nuances of computerization and its effects on people may find office automation creating a backlash of resentment, jealousies, and resistance.(m)* Obviously, managers should build a clear concept of the kind of environment they want to build in the office. They must determine whether the needs and interests of the individual or the whole organization should be met. Consequently, the real success or failure of this working environment has little to do with the hardware. Managers must treat employees as the organization's most important asset.

(f)* Connell, pp. 51-64.

(g)* McNitt, pp. 50-52.

(h)* "Potential Health Hazards".

(i)* Thackray, pp. 94-102.

(j)* McNitt, pp. 50-52.

(k)* Thackray, pp. 94-102.

(l)* Martin, pp. 42-44.

(m)* Martin, pp. 42-44.

REFERENCES

- Cakir, A., et al. Visual Display Terminals. Chichester: John Wiley and Sons, 1980.
- Connell, John J. "White Paper: First in a Series." Modern Office Procedures, March 1982, pp. 51-64.
- Kuzela, Lad. "Confronting Computer Phobia." Industry Week, 19 Sept. 1983, pp. 91-92.
- Martin, Alexia. "The Human Connection: A Strategy for Making Automation Work." Administrative Management, Feb. 1982, pp. 33-35, 42-44.
- McNitt, Jim. "Making Computers and People Compatible." Nation's Business, March 1984, pp. 50-52.
- Schraml, Mary L. "The Psychological Impact of Automation on Library and Office Workers." Special Libraries, April 1981, pp. 149-155.
- Steinbrecher, David. "Automation Outlook." Office Administration and Automation, May 1984, p. 8.
- Thackray, John. "White-Collar Blues." Management Today, March 1980, pp. 94-102.
- U.S. Department of Health and Human Services. An Investigation of Health Complaints and Job Stress in Video Display Operations. Cincinnati, Ohio: 1979.
- U.S. Department of Health and Human Services. Potential Health Hazards of Video Display Terminals. Cincinnati, Ohio: June 1981.
- "VDTs -- A New Social Disease." The Harvard Medical School Health Letter, April 1983, p. 1-5.
- "Weighing Word Processing Work Promotes Productivity." Modern Office Procedures, Oct. 1981, p. 54-57.

Microcomputers: The Growing Need
and Application in Special Education

Patricia Miller
Elementary Education Major
Northwestern College
3003 North Snelling Avenue
St. Paul, MN 55113

Computers are growing in popularity in the special education field. Despite this, many current special education teachers lack a functional knowledge of basic computer skills. In order for these teachers to keep abreast, they must develop a basic understanding of computer technology and learn to incorporate it into their teaching techniques.

Is there a place for microcomputers in special education today? Stallard says, "One of the more important areas now being investigated looks at how technology can enhance both the instructional processes of schools and the learning processes of individual children".*(6) Currently, school boards are showing much interest in having computers in the classrooms because the teacher can teach more students, allowing the school to operate on a tighter budget, and also because of the shortages of teachers in some fields (the special education field, for example). Teachers are also interested in having computers in the classroom because they can keep more efficient records, more information on each student, and the computer does some of the tedious tasks for them. Students enjoy the computers because there is more one-on-one instruction, more variety of subjects, a more relaxed atmosphere, and they can work at their own pace.

Although computer technology is present today, many teachers aren't trained in the use of this technology. This is why many colleges are requiring computer training for graduation. Teachers are being presented with several different types of software that are presently available. Basically, the software will fall into one of the following groups devised by Budoff and Hutton in 1982.

DRILL AND PRACTICE

Currently this is the most common type of software used by special education teachers. This software is designed to integrate previously learned material by presenting the student with questions which the computer either confirms as right or wrong. If the student is wrong, the computer will take him through an instructional phase and then present him with another problem.

TUTORIAL

This software program becomes the teacher and presents the lesson to the student. This software is designed in a step-by-step process in order to guide the student through the lesson. Usually this software lesson contains problems and perhaps self-quizzes to reinforce the previously learned material.

EDUCATIONAL GAMES

This software has a format that is similar to that of Pac-Man, Frogger, or any other game. In order for the student to win the game, however, he must answer questions that deal with math, spelling, or another subject area.

SIMULATIONS

This software is designed to vicariously put the student into an actual situation in order to learn from it. An example would be a program about driving. On the screen the student would see a road ahead of him. The student would drive the car down that road, following all the laws.

PROBLEM SOLVING

In this software, each program is designed to deal with one particular area of study. An example would be math. The student would be presented with a math problem; the computer would confirm whether the student's answer was right or wrong. If the student was wrong, the computer would take the student through an instructional phase and then present another question.

COMPUTER MANAGED INSTRUCTOR (CMI)

"CMI allows the teacher to use the computer as a tool in diagnostic, prescriptive and evaluative tasks."**(2) This software allows the teacher to evaluate the teaching methods being used, generate new tests and record grades more efficiently and more easily.

Special education teachers have placed a considerable amount of emphasis on success, success meaning the number of correct responses. When educators look for software that will suit their teaching units, they look for software which will allow a high level of success along with a high level of improvement by the students.

One type of software that has raised much interest is the ARC-ED courseware. ARC-ED (educational courseware which contains many characteristics of video games in order to present educational material) also features success. Success here is not measured by the number of correct responses; but rather, the amount of improvement.

"The primary characteristic of ARC-ED courseware is that it is designed to take advantage of students' interests (wants) and to motivate them to learn what they need".***(3)

There are many advantages to using computers in the special education field today. Computers can teach students in a non-threatening manner. Many students are timid or afraid of their teachers. The computer helps to ease the student into the classroom, with less fear or anxiety.

*Charles K. Stallard, "Computers and Education for Exceptional Children: Emerging Applications," Exceptional Children (Oct. 1982), p. 103.

**Milton Budoff and Leah R. Hutton, "Microcomputers in Special Education: Promises and Pitfalls," Exceptional Children (Oct. 1982), p. 126.

***Jerry D. Chaffin, Bill Maxwell, and Barbara Thompson, "ARC-ED Curriculum: The Application of Video Game Formats to Educational Software," Exceptional Children (Oct. 1982), p. 177.

Computers can also be used as a device for motivation. Students can be rewarded for good behavior or good improvement at a particular skill. This reward could be a certain amount of time playing the student's favorite game on the computer. Computers can also be used to help the student develop his fine motor skills and also his eye-hand coordination. This can be done by the student playing games that utilize the joy stick, paddles, or certain keys on the keyboard. Older students who suffer from learning disabilities can use computers as typewriters to compensate for poor motor coordination resulting in unreadable penmanship. Also, typing when used with a spelling program can enable them to write assignments and papers that they could not otherwise do.

Another benefit of the computers is that of giving the student experience with programming. "According to some, such activity encourages children to learn about how they themselves think and learn - a high level 'meta - cognitive' experience is not even shared by many adults".*(1)

Although the computers are a good way to instruct, a good teaching aid, and a motivational tool, educators must be careful as to how the computers are used in the classroom and also as to what software is used. A potential problem with using microcomputers and software for special education classes is that many of the programs require an average reading ability which would greatly hinder their capability to use the micros; therefore, this should be a major consideration in selecting programs to be used for individual students.

"The computer plays a major role in our lives and our students' lives; we must understand its potential and uses, recognizing its current limitations".**(5)

It is imperative that teachers become knowledgeable in the uses of computers. The industries are becoming more developed and advanced. They are producing many more advanced programs, and are expecting that the teachers and users know the basic skills that are needed to operate the computers. Parents in the business world are using computers and their children are using the home computers for games. College graduates are going into the education field with this computer knowledge; thus, these new graduates will be more advanced in this area than the present teachers. School boards are presently forcing computers into the schools and they are also forcing the use of the computers. Teachers may even be evaluated by their knowledge and use of the computers. Computers are here to stay. If the present teachers don't start learning and using the computers and software, they will be left behind.

*Randy Elliot Bennett, "Applications of Microcomputer's Technology to Special Education," Exceptional Children (Oct. 1982), p. 110

**Budoff and Hutton, p. 124.

BIBLIOGRAPHY

1. Bennett, Randy Elliot. "Applications of Microcomputer's Technology to Special Education." Exceptional Children (Oct. 1982), pp. 106-112.
2. Budoff, Milton, and Leah R. Hutton. "Microcomputers in Special Education: Promises and Pitfalls." Exceptional Children (Oct. 1982), pp. 123-128.
3. Chaffin, Jerry D., Bill Maxwell, and Barbara Thompson. "ARC-EDCurriculum: The Application of Video Game Formats to Educational Software." Exceptional Children (Oct. 1982), pp. 173-178.
4. Hofmeister, Alan M. "Microcomputers in Perspective." Exceptional Children (Oct. 1982), pp. 115-121.
5. Ragghianti, Suzanne, and Rosemary Miller. "The Microcomputer and Special Education Management." Exceptional Children (Oct. 1982), pp. 131-134.
6. Stallard, Charles K. "Computers and Education for Exceptional Children: Emerging Applications." Exceptional Children (Oct. 1982), pp. 102-104.